Mobility-enabled Networking Scheme for Rich Media Internet of Things Platforms

Anderson Augusto Simiscuka

A Dissertation submitted in fulfilment of the requirements for the award of
Doctor of Philosophy (Ph.D.)

Faculty of Engineering and Computing,
School of Electronic Engineering
Dublin City University

Supervisor: Dr. Gabriel-Miro Muntean

January, 2020

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, and that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: _____ ID No.: _____ Date: _____

Acknowledgements

Firstly, I would like to thank my supervisor Dr. Gabriel-Miro Muntean for his support before and during my PhD years. I cannot express in words my gratitude for all your patience, enthusiasm, advice, for helping me many times until late hours so we could meet paper deadlines, and for "promoting" me from an intern in the lab to a PhD candidate. I do not only see you as a supervisor, but also as a mentor and friend!

I want also to thank Dr. Cristina Hava Muntean and Dr. Sebastian Robitzsch, as through them I became part of the DCU's Performance Engineering Lab team, which led me to develop this research.

A big thanks to my friends in the lab, especially to John Monks, Fabio Silva, Ting Bi, Mohammed Togou, Longhao Zou, Tejas Markande, Danielle Sheridan, Sami Abbas and Ian Kavanagh. I was lucky to work with great people and I am grateful for the support, ideas, discussions and laughs during the good and hard times of the PhD.

I am also lucky to have great friends who were always with me during the past few years, in Ireland and in Brazil, so I need to thank some of them: Fernanda, Killian, Fatima, Orla, Jen, Roberta, Monica, Jhonata, Barbara, Alinie, Amanda, Diego, Erickson and Andrea.

My eternal gratitude to my dear parents Maria Zilda and Emilio and to my brother Lucas, who although are physically far away from me, are constantly by my side, supporting me and praying for me.

Finally, thanks to my examination panel, Prof. David Gómez-Barquero, Dr. Kevin McGuinness and Dr. Marissa Condon. Thank you all for your time and allowing this to happen.

This work has been supported by the Irish Research Council Entrerprise Partnership Scheme and Dublin City University, grant number EPSPG/2015/178 and in part by Science Foundation Ireland grant 13/RC/2094 to Lero – the Irish Software Research Centre (http://www.lero.ie) and the European Union's Horizon 2020 Research and Innovation programme under grant no. 688503 for the NEWTON project (http://newtonproject.eu).

Publications

Journals

- A. A. Simiscuka, T. M. Markande and G.-M. Muntean, 'Real-Virtual World Device Synchronisation in a Cloud-enabled Social Virtual Reality IoT Network', *Published in IEEE Access,* Volume 7, 2019, pp. 106588–106599.
- A. A. Simiscuka and G.-M. Muntean, 'An Innovative Relay-based Architecture for Performance-oriented Communications between Mobile IoT Devices', *Under review for IEEE Internet of Things Journal*.
- A. A. Simiscuka, R. Trestian and G.-M. Muntean, 'Getting Social with Things: A Survey on Rich Media IoT Applications, Technologies, Challenges and Open Issues', *Under review for IEEE Communications Surveys and Tutorials*.

Conferences

- A. A. Simiscuka, C. Muntean and G.-M. Muntean, 'A Networking Scheme for an Internet of Things Integration Platform', *IEEE International Conference on Communications (ICC'17) Workshop*, Paris, France, 2017, pp. 1–6.
- A. A. Simiscuka, M. Bezbradica and G.-M. Muntean, 'Performance Analysis of the Quality of Service-aware Networking Scheme for Smart Internet of Things Gateways', *IEEE International Wireless Communications and Mobile Computing Conference (IWCMC'17)*, Valencia, Spain, 2017, pp. 1370–1374.
- A. A. Simiscuka and G.-M. Muntean, 'Synchronisation between Real and Virtual-World Devices in a VR-IoT Environment', *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB'18)*, Valencia, Spain, 2018, pp. 1–6.

- A. A. Simiscuka and G.-M. Muntean, 'A Relay and Mobility Scheme for QoS Improvement in IoT Communications', *IEEE International Conference on Communications (ICC'18) Workshop*, Kansas City, USA, 2018, pp. 1–6.
- A. A. Simiscuka and G.-M. Muntean, 'Age of Information as a QoS Metric in a Relay-Based IoT Mobility Solution', *IEEE International Wireless Communications and Mobile Computing Conference (IWCMC'18)*, Limassol, Cyprus, 2018, pp. 868–873.
- T. Bi, A. A. Simiscuka, F. Silva, M.A. Togou, G. Rozinaj and G.-M. Muntean, 'A Study of Learning Experience with a DASH-Based Multimedia Delivery System', *International Conference on Education and New Learning Technologies (EDULEARN'18)*, Palma de Mallorca, Spain, 2018, pp. 1–6.
- S. Abbas, A. A. Simiscuka and G.-M. Muntean, 'A Platform Agnostic Solution for Inter-Communication between Virtual Reality Devices', *IEEE World Forum on Internet of Things (WF-IoT'19)*, Limerick, Ireland, 2019, pp. 1–6.
- D. Sheridan, A. A. Simiscuka and G.-M. Muntean, 'Design, Implementation and Analysis of a Twitter-based Social IoT Network', *International Symposium on Sensors and Instrumentation in IoT Era (ISSI'19)*, Lisbon, Portugal, 2019, pp. 1–6, Awarded Best Student Paper.

Contents

List of Tables

List of Figures

List of Acronyms

3GPP – 3rd Generation Partnership Project 3D - Three Dimensional
6LoWPAN – IPv6 over Low-power Wireless Personal Area Networks
ACF – Auto Correlation Function
ADR – Adaptive Data Rate
AES – Advanced Encryption Standard
ALE – Application Level Events
AMQP – Advanced Message Queuing Protocol
ANOVA – Analysis of Variance
AOMDV – Ad-hoc On-demand Multipath Distance Vector
AODV – Ad-hoc On-Demand Distance Vector
AoI – Age of Information
AP – Access Point
API – Application Programming Interface
AR – Augmented Reality
BATMAN - Better Approach To Mobile Ad-hoc Networking
BI – Business Intelligence
BLE – Bluetooth Low Energy
BS – Base Station
CBC-MAC – Cipher Block Chaining Message Authentication Code
CC – Consistency Check
CCM – Counter with CBC-MAC
CCTV – Closed-Circuit Television
CoAP – Constrained Application Protocol
CRM – Customer Relationship Management
CSMA-CA – Carrier Sense Multiple Access with Collision Avoidance
CPU – Central Processing Unit
CSS – Chirp Spread Spectrum
D2D – Device-to-Device

DAD – Duplicate Address Detection
DDoS – Distributed Denial of Service
DDS – Data Distribution Standard
DIO – DODAG Information Object
DL – Downlink/Download
DNP3 – Distributed Network Protocol
DNS – Domain Name System
DNS-SD – DNS Service Discovery
DODAG – Destination Oriented Directed Acyclic Graph
DoS – Denial of Service
DSR – Dynamic Source Routing
DSSS – Direct Sequence Spread Spectrum
DTLS – Datagram Transport Layer Security
EPC – Electronic Product Codes
ER – Edge Router
ERP – Enterprise Resource Planning
FFD – Full Function Device
FSK – Frequency Shift Keying
FTP – File Transfer Protocol
GPIO – General Purpose Input/Output
HD – High Definition
HDFS – Hadoop Distributed File System
HDMI – High-Definition Multimedia Interface
HTTP – Hypertext Transfer Protocol
ICT – Information and Communications Technologies
IEEE – Institute of Electrical and Electronics Engineers
IETF – Internet Engineering Task Force
ILP – Integer Linear Programming
IoT – Internet of Things
IoHT – Internet of Health Things
IoMT – Internet of Mobile Things
IPC – Inter-Process Communication
IQR – Interquartile Range
ISM – Industrial, Scientific, and Medical radio band
ITINP – IoT Integration Platform

ITU – International Telecommunication Union
JSF – JavaServer Faces
JSON – JavaScript Object Notation
JVET – Joint Video Exploration Team
KIM – Key Identifier Mode
KPI – Key Performance Indicator
LAN – Local Area Network
LED – Light-Emitting Diode
LoRa – Long Range
LoRaWAN – Long Range Wide-Area Network
LoWPAN – Low-Power Wireless Personal Area Networks
LR-WPAN – Low-Rate Wireless Personal Area Networks
LTE – Long-Term Evolution
LTE-A – LTE Advanced
LTE-M – LTE Category M1
M2M – Machine-to-Machine
MAC – Media Access Control
MBPP – Multiple Base station and Packet Priority-based Clustering scheme
mDNS – Multicast DNS
MIMO – Multiple-Input and Multiple-Output
MIoT – Multimedia Internet of Things
MPEG – Moving Picture Experts Group
MPTCP – Multipath TCP
MQTT – Message Queue Telemetry Transport
MQTT-SN – MQTT for Sensor Networks
MR – Mixed Reality
MTC – Machine Type Communication
NDEF – NFC Data Exchange Format
NETSMITS – NETworking Scheme for sMart IoT gatewayS
NFC – Near Field Communication
NS-3 – Network Simulator 3
NTP – Network Time Protocol
OLSR – Optimized Link State Routing Protocol
OMG – Object Management Group
OS – Operating System

P2P – Peer-to-Peer
PDF – Probability Density Function
PDV – Packet Delay Variation
PHY – Physical Layer
PTM – Point-to-Multipoint
PTP – Precision Time Protocol
QoE – Quality of Experience
QoS – Quality of Service
QR – Quick Response
RAM – Random Access Memory
REMOS-IoT – RElay and MObility Scheme for the Internet of Things
REST – Representational State Transfer
RF – Radio Frequency
RFD – Reduced Function Device
RFID – Radio-Frequency Identification
RN – Relay Node
ROM – Read-Only Memory
RPL – Routing Protocol for Low Power and Lossy Networks
RSM – Response Surface Methodology
RSSI – Relative Received Signal Strength
RSRP – Reference Signal Received Power
RTP – Real-time Transport Protocol
RTCP – Real-Time Control Protocol
SCA – Side-Channel Attacks
SCADA – Supervisory Control And Data Acquisition
SCTP – Stream Control Transport Protocol
SDK – Software Development Kit
SG – Smart Gateway
SHA – Secure Hash Algorithms
SIoT – Social Internet of Things
SLA – Service Level Agreement
SNR – Signal-to-Noise Ratio
SoA – Service-Oriented Architecture
SoC – System-on-a-Chip
SQL – Structured Query Language

TCP – Transmission Control Protocol
TDMA – Time-Division Multiple Access
UAV – Unmanned Aerial Vehicles
UDP – User Datagram Protocol
UI – User Interface
uID – Ubiquitous ID
UL – Uplink/Upload
URL – Uniform Resource Locator
UWB – Ultra-Wideband
V2X – Vehicle-to-Everything
VANET – Vehicular Network
VEoT – Virtual Environment of Things
VM – Virtual Machine
VR – Virtual Reality
VRITESS – VR-IoT Environment Synchronisation Scheme
VRITIP – VR-IoT Platform
WMN – Wireless Mess Network
WSN – Wireless Sensor Network
WWW – World Wide Web
XML – Extensible Markup Language
XMPP – Extensible Messaging and Presence Protocol

Abstract

Mobility-enabled Networking Scheme for Rich Media Internet of Things Platforms

Anderson Augusto Simiscuka

This thesis proposes and evaluates an Internet of Things (IoT) solution consisting of architectures and algorithms, which enable several heterogeneous devices to inter-communicate, providing support for their offered services. An IoT network needs to enable exchange of diverse data types including rich-media and sensor data and handle a large number of devices, offering services at high quality levels. These goals are achieved by innovative solutions, which improve the performance and quality of IoT services and allow exchange of rich-media content. The proposed architecture consists of IoT objects and their services, smart gateways, supporting IoT object network connectivity, and cloud platforms employed for resource management.

The performance and quality of IoT services are improved through NETSMITS (NETworking Scheme for sMart IoT gatewayS), which contains an algorithm that uses Quality of Service (QoS) and service relevance metrics in order to efficiently cluster inter-communicating IoT objects, attach them to the most suitable smart gateway and improve the performance of their services. The IoT mobility challenges are addressed through REMOS-IoT (RElay and MObility Scheme for the Internet of Things), which introduces algorithms that use QoS and service relevance metrics, age of information, and objects location in order to provide the best connectivity for devices and achieve better overall performance and QoS. A higher layer in the architecture contains VRITESS (VR-IoT Environment Synchronisation Scheme), which allows virtualisation of the IoT devices to fulfil requirements of certain rich-media devices and platforms, such as virtual reality (VR) devices and social media, integrating them to the IoT network.

The contributions presented in this thesis include simulation models and testbeds for the mobility-enabled networking scheme for rich-media IoT, demonstrating performance improvements in comparison with other baseline solutions. A survey of IoT architectures, applications, QoS, protocols and mobility solutions is also provided.

Chapter 1

Introduction

1.1 The Internet of Things: A Networking Revolution

The recently introduced Internet of Things (IoT) paradigm refers to the creation of an integrated network that connects diverse devices, mostly personal and consumer-based, enabling them to communicate, synchronise, learn and act in such manner in order to optimise their functionality and meet certain goals in terms of service quality and performance, user preference, energy consumption, etc. IoT aspects complement the Smart Home approach (i.e. referred here as Smart Business), which can be defined as a residence/business premise equipped with computing and information technology which responds to the needs of the occupants, working to support their needs. However, no operations are performed in isolation and the key aspect is to interconnect these Smart Businesses and Homes and create a real large scale IoT, enabled to respond to user needs regardless of what "Thing" has acquired the data sought and where it is located in timely manner and at high quality. Figure 1.1 presents how these things connect with each other in a premise and with other devices in different locations through a cloud-based connection.

A large scale IoT is going to be composed of billions of devices, such as sensors, smart devices, appliances, wearables, etc. In the past decade, sensors prices have dropped by 50%, bandwidth-related prices have dropped by more than 97% and processing prices have dropped by more than 98% [1]. These and other recent technological advances in relation to devices, networks, protocols and applications have enabled IoT to offer a wide range of services and is expected to grow to over $3 trillion annually by 2026 [2]. Thanks to these advances, 25 billion devices are expected to be IoT networked by 2020. There are many challenges in connecting such a large number of devices, i.e. maintain good user perceived Quality of Service (QoS) levels

and low energy consumption. Efficient data transmission is another challenge in heterogeneous networks that handle small and large data at the same time i.e. sensor data and high definition videos [3]. It is also expected from IoT solutions that they provide insights and deliver customised experiences for users. Therefore, another challenge is the creation of a smart decision-making network.

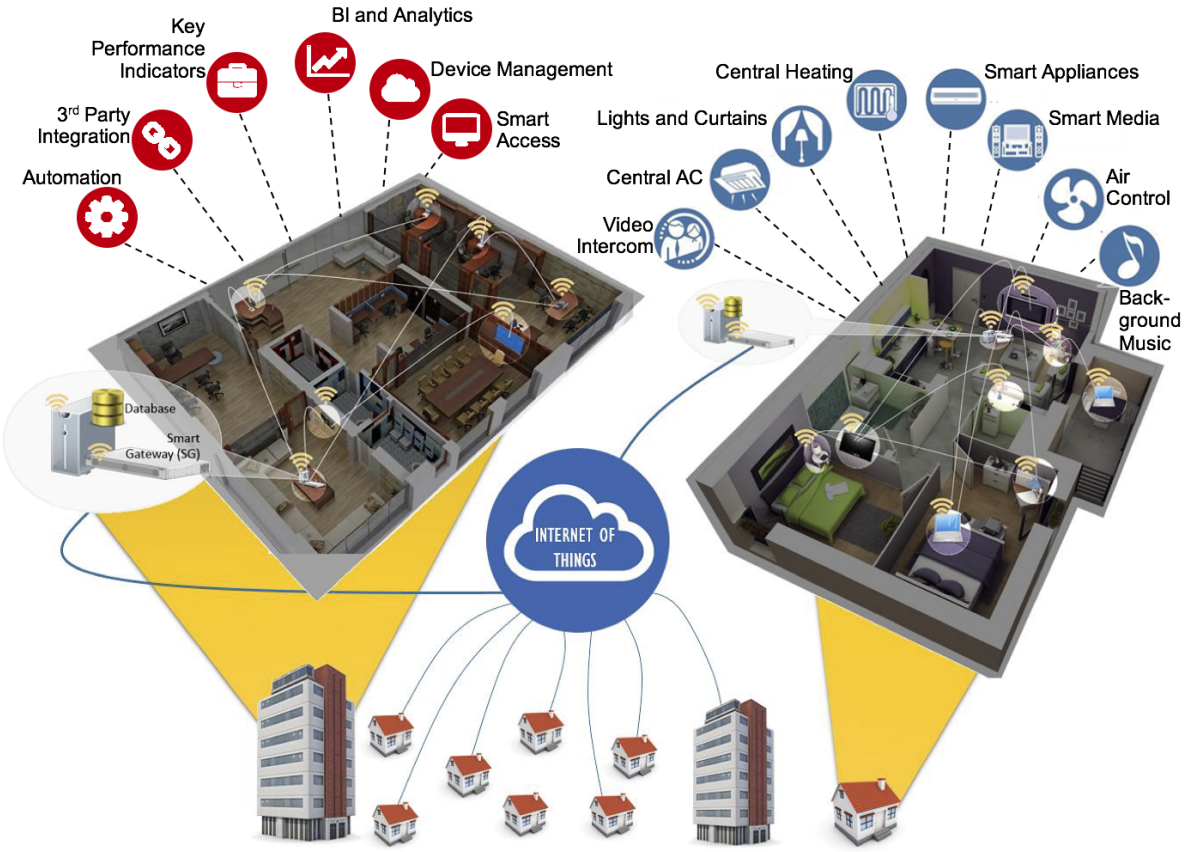


Figure 1.1: Internet of Things Features for Smart Homes and Businesses

As seen in Figure 1.1, smart homes and businesses benefit from IoT in several activities [4], such as:

- Automation: triggers, alerts and notification tasks in real time;
- 3rd Party Integration: integrations with Customer Relationship Management (CRMs), Enterprise Resource Planning (ERPs) and team collaboration tools;
- Key Performance Indicators: productivity, efficiency and uptime tracking;
- Business Intelligence (BI) and Analytics: customisable analytics and insights which turn data into information and business value;

- Device Management: provisioning, monitoring and management of devices from a set of controls;
- Smart Access: tracking of smart businesses from portals, apps and Application Programming Interfaces (API).

Smart Homes also have received several IoT applications [5], such as:

- Video Intercom: stability to run indefinitely and adapt to variations in WiFi bandwidth;
- Central Air Conditioning: control air conditioning from smartphones;
- Lights and Curtains: control brightness, colour of lights using smartphones and tablets, motorised curtains can automatically adjust themselves based on daylight, saving energy;
- Central Heating: control thermostats using smartphones;
- Smart Appliances: smart automated vacuum cleaners, smart fridges, etc.;
- Smart Media: Amazon Echo, Google Home, virtual reality (VR), smart TVs, gaming consoles;
- Air Control: automated ventilation control of environments;
- Background Music: wireless speaker systems.

The applications of IoT devices go beyond homes and business. Different solutions are being introduced to agriculture (soil monitoring sensors, drones for field monitoring), health (wearable sensors, patient flow), manufacturing (automation, maintenance alerts, production flow monitoring, VR/Augmented Reality (AR) prototyping), cities (smart parking, vehicular networks, smart lighting), etc.

The different types of content, connections, data, nature of devices and the lack of standards, resulted in several different approaches, protocols and technologies from the IT industry and academia.

Some devices such as smartphones, for example, contain several of the technologies presented above, and multiple ways of connecting to available networks.

The massive amount of devices and services available in IoT networks, which involve not only smart homes and businesses, but also manufacturing, farming, transportation, health services, and so on, bring numerous challenges for researchers and developers. There is a need for algorithms and solutions that provide the best connectivity to devices and users considering the available networks, content, location, etc. There is also a need for devices to be accessible and user-friendly.

Based on the features offered by IoT networks, research challenges can be identified before solutions are proposed. These challenges and research gaps are presented next.

1.2 IoT Research Challenges

Numerous reports predict that by 2020 more than 20 billion devices, both mobile and fixed, will be connected in the IoT. An IEEE report [6] identified areas of research that focus on the challenges brought by the new innovation paradigms of IoT, especially regarding mobility, which originated the term Internet of Mobile Things (IoMT). Examples of devices in the IoMT include robots, drones, cars, buses or trains, wearable devices (such as smart watches and wrist bands) and smartphones.

There are many challenges in keeping a great number of devices connected to the Internet, especially when many of them are moving around.

Regarding **mobility and network design**, mobile IoT devices need to be able to connect with each other and other devices of the IoT network, adding the constraints of location, intermittent connectivity and RF (Radio Frequency) link variability. Internet-connected cars, for example, need to transmit and receive data from different gateways depending on their location.

In fixed deployed sensing and actuation platforms, normally devices are aware of their locations, have synchronised clocks, have awareness of their neighbouring devices, configure a consistent set of parameters for energy saving and security (such as consistent sleep and wakeup schedules, appropriate power levels of communication, and security keys). When mobility is added, the topology of the system will be more volatile. Challenges regarding **robustness** include device discoverability, power usage optimisation, communication protocols, and maintainability of a long-lived dynamic system.

In a network that includes self-driven cars and drones that will deliver goods and perform other activities, it is crucial that programmers do not have to manually decompose each device's goal into a set of single tasks. Real-time **coordination** among mobile sensing and actuation platforms is an important research challenge that needs to be addressed.

In sensor networks, network communication and input/output operations need to occur in parallel with data processing. **Concurrency** is a challenge in IoT due to the number of connected devices and, especially for devices with mobility, because of the unpredictability of their navigation time and movement patterns, creating areas with

a great number of devices at the same time, such as sports events, concerts, traffic jams, etc.

The understanding of **mobility patterns** (e.g. different types of vehicles may have pre-defined or specific patterns) is important for the identification of suitable mobile sensors to install on the device, car, etc.; which protocols are most suitable; which types of sensing platforms can interact with each other; the need for gateways, etc.

The large number of sensors, devices and data generated in an IoT network makes necessary the aggregation of sensing streams and data to an integrated interface to other services and functionalities, such as road conditions with information from multiple vehicle sensors. **Data aggregation** and its implementation, however, need to consider which and how sensor data can be aggregated according to mobility patterns, within and outside the network range.

Due to the enormous amount of data generated in an IoT network, it usually needs to be filtered and/or processed, however when location and time are added, the **filtering and processing** need to be carefully done and different approaches need to be introduced.

Regarding **data storage**, the massive volume of IoT data needs to be stored in an optimal way for viewing and future analysis considering network topology, latency, mobility patterns, location and synchronisation.

Actuation enables users to modify IoT networks so the devices can, for example, self-repair, changing their states, configuration, aggregate data, etc. Actuations for IoT resources, especially mobile ones, are complex, and may require service compositions, linking services to actuation commands. The service composition, however, may affect QoS regarding pre-defined service-level agreements (SLAs).

Opportunistic computing, which is how users and their mobile devices can interact with others opportunistically, brings challenges in discovering the optimal solutions regarding decentralised interactions among the users and the network.

IoT is usually perceived as a large network of sensors and actuators, which indeed are an important part of IoT; however, there is a myriad of other "more complex" devices that belong to the realm of IoT. Smart TVs, smartphones, VR headsets, CCTV, in-built displays of all sorts, touchscreens, handhelds, smart watches, drones with screens, smart billboards, etc., are devices that belong to the IoT and cooperate with sensors and other devices to bring multimedia-based services and applications to users. These heterogeneous media-rich devices, however, present different challenges and opportunities to researchers. Multimedia traffic and sensing, data storage,

social aspects, privacy and security are fields that need to be studied in the rich media IoT, in order to allow high quality and meaningful communications in such diverse and complex networks.

Unfortunately, many IoT devices are still complex to use and do not offer a mature and simple experience to users [7]. Beyond the employment of multimedia solutions, the use of VR technology is a powerful avenue to bring users a rich media experience and an increase sense of reality, by offering a realistic, visual, auditory, and tactile sense of the world. The possibility of operating virtual objects using human senses, language, and gestures, can make it easier for users to understand and interact with the IoT devices [8]. Devices located in extreme environments (e.g. landslide sensors, water level monitoring devices) or which are not very user-friendly, can be manipulated in a simpler way in a virtual environment.

The opportunities and research challenges raised in this section served as inspiration for the construction of the ideas that will be presented in this thesis. It is fair to say that IoT currently needs more research and development towards improved network performance and user experience, and the goals of this thesis in contributing to this fields will be described in the next section.

1.3 Motivations and Objectives

While the IoT brings many new devices with exciting possibilities, it is necessary to understand that innovative solutions need to optimise performance and the way devices interact with each other and connect to the Internet. In order to address the research challenges outlined on section 1.2, this thesis presents an architecture acting as an umbrella of IoT solutions that improve performance and integrates IoT devices, while allowing users to operate these devices in a user-friendly manner.

One of the components of the proposed architecture is the novel large-scale **IoT Integration Platform (ITINP)**, designed along with the other contributions of this thesis. ITINP is a cloud-based server, which leverages IoT technology to integrate and synchronise the operation of various user client devices, mostly multimedia enabled, will be used in order to improve user Quality of Experience (QoE) in a Smart Business/Home environment. ITINP relies on a two-stage architecture: local and global. Locally, at the level of each Smart Business/Home, ITINP makes use of a Smart Gateway (SG), which performs local data management, storage, processing and dissemination and acts as a connection point with the remaining IoT. Globally,

ITINP operation is based on SGs of several Smart Businesses/Homes interacting with each other and helping locate the information sought from the best source in terms of the performance associated with its potential delivery to the requesting user.

ITINP is important for allowing development of solutions that address performance issues of IoT. However, another aspect that still needs to be covered is the accessibility of devices, as users require not only a reliable network, but also one with friendly interfaces.

The contributions of this thesis also make use of the novel **VR IoT Platform (VRITIP)**, which was designed to deploy the architecture proposed in this thesis. VRITIP is a solution responsible for representing real devices in a virtual environment, presenting to users a user-friendly virtual version of IoT devices, and also for keeping these virtual devices updated with the latest alterations made in the real world. VRITIP sends updates made in the virtual world to ITINP, which controls the real world devices. The virtualised translation of IoT services in VRITIP can be accessed through interfaces that have high user acceptance, such as VR devices and social networks.

The IoT solution including ITINP and VRITIP should support the delivery of different types of data, with special focus on rich media content, which has strict requirements in terms of bitrate, loss, delay/jitter, etc. It should also aim to allow users to access the rich media content and smart devices in a user friendly manner, in order to make the IoT network accessible.

The methodology to achieve these objectives include:

1. Architectural design of the solution: Based on the advantages in term of scalability and efficiency, this project will employ an overlay on top of the heterogeneous network infrastructure as the method to address the architectural design. Challenges include identifying and implementing the most suitable type of IoT architecture for the large-scale network, considering there are various existing layered architectures. The architecture also needs to offer a seamless user experience for IoT devices, and a synchronised representation of IoT devices in a virtual world.

2. Extending existing network models to support IoT-based smart networking and increase number of inter-communicating devices: Existing models consider network nodes equipped with the whole networking stack, which is not the case for many IoT-enabled devices, which are envisaged to include light networking support. This also affects scalability and the challenge here is to create models

for light network nodes and to enable communication between increased numbers of such nodes.

3. Network performance monitoring and network selection modelling: Non-invasive observation-based and probe-based solutions, respectively will be explored for network performance monitoring in terms of their suitability for IT-INP. Methods for network selection that aim at keeping the mobile users Always Best Connected anywhere and anytime are also going to be used. The main parameters considered include delivery performance (QoS), mobility metrics and number of smart objects.

4. Model simulation refers to the implementation of the solution and algorithms using an existing open-source network simulation tool, such as Network Simulator 3 (NS-3) [9], and perform simulations considering various network types/ technologies, diverse network conditions, increasing number of devices and various content.

5. Prototyping and testing refers to implementation of in-lab testbeds and simulations in order to assess the effect of the proposed algorithms on QoS. This phase also includes the implementation of user interfaces and applications for visualisation, virtualisation and manipulation of smart devices, which can include integration with technologies such as VR and social media.

1.4 Problem Statement

ITINP and VRITIP need to be used for the implementation of solutions that address the challenges mentioned in section 1.2. Considering the methods presented in this section, it is possible to formulate questions regarding which issues and challenges the contributions presented in this thesis should focus on. Several aspects of IoT must be considered in order to maintain high quality of service, support mobility and increase quality of user experience:

1. The high number of devices in IoT networks and the distribution of several gateways using multiple technologies to access the Internet are a challenge for a large unified IoT network. The heterogeneity of devices and the distribution of services in the network are additional challenges, because certain devices can

affect the performance of the network while other devices may not be connected to the optimal gateway at a certain time.

Therefore, a solution must make decisions on devices affecting the network. For these decisions, such as disconnecting a device, re-attaching it to another gateway, etc., devices QoS metrics must be monitored, so their behaviour in the network can be analysed. Metrics that demonstrate which devices and services are performing poorly must be identified, and algorithms must organise devices in a smart manner, allowing an increase in the number of connected devices. The solution also needs to consider that multiple gateways are available and some might be overloaded, while others not.

2. In relation to QoS, a number of solutions can improve the values of some metrics during data delivery, such as delay, packet loss and throughput. In order to assess and enhance QoS, solutions employ several approaches, including adaptation schemes, scheduling techniques, clustering algorithms and other innovative communication approaches such as relaying and device-to-device communications.

   Many IoT devices are constantly moving, and there is a need for solutions that provide connectivity when these devices cannot reach any base station/access point or there is limited bandwidth available. The use of metrics for detecting if devices are moving away from a gateway must be considered, so that the solution can proactively offer connectivity alternatives for mobile devices.

3. Regarding user experience in IoT, many devices such as smart appliances, wearables, health monitors, smart cars, etc. provide interfaces for user input. However, certain IoT devices are still complex to operate and do not provide a simple and mature user experience. Other IoT objects such as landslide sensors or devices for water level monitoring, can be located in hazardous places.

   Solutions must be developed allowing users to enjoy a rich media and friendly experience, while consuming local and remote IoT services. In the realm of rich media solutions, VR and social media technologies are available, offering intuitive user interfaces.

   Integrating rich media technologies to IoT also brings new challenges: How can devices be synchronised to the virtual representations displayed to users? How often should data be updated and refreshed? Which technologies can be used

or adapted for remote communications with low latency?

The ultimate goal must be to enable users to easily consume content and share access to services and devices.

1.5 Thesis Contributions

In order to present solutions for the issues identified in the previous section, the mechanisms that are part of these solutions need to be selected. These mechanisms must focus on improving device performance, allowing an increase on the number of devices in the network, and benefiting users with low-latency interfaces.

Therefore, the solutions proposed in this thesis include a clustering scheme, allowing devices to be re-clustered according to performance metrics and device relation within the network; a mobility-enabled solution, providing support to mobile devices with relay nodes and location metrics; and a synchronisation solution for real-virtual device integration, allowing IoT devices to be operated with rich media interfaces such as VR and social media.

Aiming to fulfil the challenges, motivations and objectives presented in the previous sections, the contributions of this thesis include the proposals of the following IoT schemes:

1. NETSMITS (NETworking Scheme for sMart IoT gatewayS), which provides smart-clustering of IoT objects, attaching them to the most suitable smart gateway in order to improve the performance and quality of their services. Gateways are interconnected by a major cloud-based Internet of Things (IoT) Networking Platform. NETSMITS brings the novelty of employing and assessing the values of QoS and Relevance metrics at the level of devices and gateways connected to the cloud-deployed IoT Integration Platform (ITINP), in order to perform smart clustering of devices. NETSMITS benefits include support to an increased number of inter-communicating IoT objects while maintaining higher quality of their offered services. Results demonstrate that NETSMITS is able to avoid packet loss, allow highest throughput per device to be achieved and reduce delay by 85% by rearranging devices in the available gateways. It also enables 63% more devices to be connected to the better performing gateway.

2. REMOS-IoT (RElay and MObility Scheme for the Internet of Things) works along with NETSMITS to provide mobility support in the devices, while also

using other devices for relay to maintain device connectivity when smart gateways are out of reach. REMOS-IoT contains novel algorithms and records scores for QoS, relevance, age of information and location, in order to cluster IoT devices efficiently and support mobility, acting proactively in finding new possible smart gateways for devices in motion. Testing demonstrated benefits in results for uplink and downlink, which were recorded in terms of throughput, packet loss and delay. REMOS-IoT also outperforms another solution in terms of packet loss and delay.

3. VRITESS (VR-IoT Environment Synchronisation Scheme) is a novel synchronisation algorithm that allows users to seamlessly operate IoT devices, such as beacons and sensor functions in single-board computers such as the Raspberry Pi, in VR and social media environments. VRITESS benefits interactions with IoT devices, which can become more user-friendly in virtual environments, and also helps bridging the gap between rich media content and IoT, allowing social media aspects to be integrated into IoT networks. VRITESS can be executed with different communications protocols in local or remote environments and results compare local and cloud-based approaches, as well as indicate low latency in the solution. Through device virtualisation, social media can also be integrated into the virtual space. Novel algorithms for social network and IoT integration are proposed with the use of IoT-related protocols. Extra results demonstrate that social networks can be used to operate IoT devices with minimal latency.

1.6 Thesis Structure

The thesis was structured in eight chapters as follows:

- Chapter 1 – introduces an initial background of the IoT area, presenting challenges, motivations and objectives that directed the solutions presented in the following chapters. This chapter also briefly introduces the main contributions of the research work presented in this thesis.
- Chapter 2 – presents the technical background used as a foundation for the solutions presented in this thesis.
- Chapter 3 – contains a comprehensive survey of state-of-the-art research on the topics relevant for this thesis contributions: IoT architectures, protocols, rich

media IoT solutions, clustering and QoS for IoT, IoT mobility, and social interactions of IoT devices.

- Chapter 4 – introduces the overall system architecture in which the three major contributions are built on.
- Chapter 5 – details the solution design, algorithms, testing scenarios, results and analysis of NETSMITS, the NETworking Scheme for sMart IoT gatewayS.
- Chapter 6 – presents the solution design, algorithms, testing scenarios, results and analysis of REMOS-IoT, the RElay and MObility Scheme for the Internet Of Things.
- Chapter 7 – contains the solution design, algorithms, testing scenarios, results and analysis of VRITESS, the VR-IoT Environment Synchronisation Scheme for Social IoT.
- Chapter 8 – concludes the thesis and presents possible future work directions.

Chapter 2

Technical Background

This chapter introduces the technical background relevant for mobility-enabled rich media IoT solutions. Therefore, classic IoT areas such as sensor technologies, hardware and software for IoT, device identification, communications and protocols, IoT services and security are detailed. Rich media IoT, which is the main type of IoT data used in this work, is also presented, such as the relation of IoT with virtual reality, multimedia and social networks. The chapter is then concluded with a brief summary.

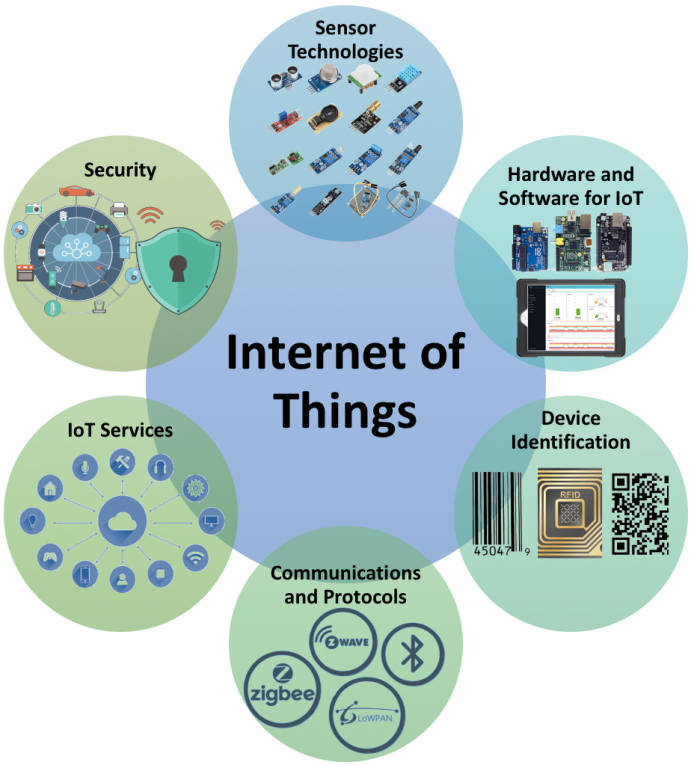


Figure 2.1: IoT Classic Areas

Table 2.1: Classic IoT Technologies

IoT Areas	Technologies		
Sensor Technologies	Smart Sensors, Wireless Beacons, Wearable Sensing Devices, Embedded Systems, Actuators, RFID, NFC		
Hardware and Software for IoT	**Hardware**	SmartThings, Systems-on-a-Chip, Smartphones	
	Software	Contiki, Cooja, NS-3, TinyOS, LiteOS, Riot OS, Android, Cloud (Google Cloud IoT Core, Hadoop), Apple HomeKit	
Device Identification	**Naming**	EPC, Ucode, QR Codes	
	Addressing	IPv4, IPv6	
Communications and Protocols	RFID, NFC, UWB, BLE, Bluetooth, LTE, 5G, LTE-M, NB-IoT, WiFi, WiFi Direct		
	Application Protocols	DDS, CoAP, AMQP, MQTT, MQTT-SN, XMPP, HTTP, REST	
	Service Discovery	mDNS, DNS-SD	
	Infrastructure Protocols	**Routing**	RPL, AODV, AOMDV, OLSR, BATMAN, DSR
		Transport	SCTP, MPTCP
		Link Layer	IEEE 802.15.4, Ethernet
		Network Layer	LoRaWAN, 6LoWPAN, ZigBee
		Link Layer	IEEE 802.15.4, Ethernet
		Physical/Device Layer	LTE-A, EPCglobal, IEEE 802.11ah, IEEE 802.15.4, Z-Wave, LoRa
IoT Services	Multimedia IoT, Smart Homes, Smart Businesses, Smart Cities, Smart Grid, Smart Healthcare		
IoT Security	IEEE 802.15.4, 6LoWPAN/6LoWPSec, RPL, CoAP, Blockchain		

2.1 Classic IoT Technologies

IoT is happening thanks to several advances in areas such as sensor technologies, hardware and software for IoT, device identification, communications and protocols, IoT services and security, as illustrated in Figure 2.1.

The next subsections will detail important IoT areas and technologies, summarised in Table 2.1, highlighting the ones that are more relevant to the rich media IoT devices.

2.1.1 Sensor Technologies

One of the main characteristics of IoT is the ability of sense real world environments. The data collected by the sensors are at the start of the process of information generation, which usually involves consolidating data in the cloud, smart gateways, cluster heads, etc.

Smart sensors, including temperature, pressure, humidity, weather, current, and voltage sensors provide data using standardised communication protocols, such as the IEEE 1451 Smart Transducer Interface for Sensors and Actuators family of standards, such as the Distributed Network Protocol (DNP3, IEEE 1815) and the Precision Time Protocol (PTP, IEEE 1588) [10].

Wireless beacons broadcast/transmit a small amounts of data. Bluetooth Low Energy (BLE) beacons can be used, for example, to give vouchers to smartphone users in shops, indoor motion in buildings, etc. Information transmitted by BLE beacons can be strings of text, Uniform Resource Locators (URLs), unique IDs and temperatures. Both Apple and Google have developed protocols for BLE beacons, the Apple iBeacon and Google Eddystone. These protocols are one dimensional and require a Broadcaster, which transmits data to its surroundings, and an Observer, which passively listens to devices in its area and processes the data from the packets it receives [11], [12].

Wearable sensing devices (e.g., smart watches and wristbands) are embedded with a number of different sensors such as air temperature sensor and humidity sensor, and non-invasive bio-sensors such as heart rate sensor, thermometer sensor, etc. They can be used in hospitals and also for ordinary users, who have gains in quality of life with the data provided by the sensors [13].

Modern low-power embedded systems also provide IoT functionality out-of-the-box due to the availability of network-aware microcontrollers and system-on-chip (SoC) solutions [14], such as Raspberry Pi, Arduino, Beaglebone and Intel Galileo.

Actuators aim to change the behaviour of the environment they are in or physical systems. Actuator nodes usually have strong computation and communication powers and longer battery life. There are different actuator middleware approaches for IoT services offering important functions for different applications such as efficient control [15].

Radio-frequency identification (RFID) allows identification and personalised services through an electronic label and a reader system. RFID and near field communication (NFC), provide services such as access to restricted areas, personal identi-

fication in purchase transactions or public transportation, with tags in cards or other objects interacting with the RFID system. IoT technologies benefit from the use of passive RFID brings in cost effectiveness and simplicity, without the need for battery usage or replacement [16].

2.1.2 Hardware and Software for IoT

The computational intelligence in IoT is present in Hardware and Software. Several companies and academia are developing solutions for smart homes, cities, businesses, etc. Hardware and software platforms, operating systems and simulators will be described in this section.

2.1.2.1 Hardware

Samsung SmartThings is an example of a mature IoT platform with a growing set of apps, the SmartApps, supporting 132 types of devices. SmartThings has similar security design principles with other frameworks, such as authorisation and authentication for device access and a built-in mechanism to protect device operations against third-party apps through so-called capabilities [17].

Devices such as Systems-on-a-Chip (SoC) (e.g. Arduino, Intel Galileo, Raspberry Pi, BeagleBone, Gadgeteer, Cubieboard) provide an integrated solution with several components on a single chip. These devices usually feature digital, analog, mixed signal, and radio-frequency components and functions, giving designers a complete set of features to create innovative solutions and solve challenges of modern electronics [18].

Smartphones are often used to control and interact with IoT devices playing a central role in future IoT environments. Examples of devices that can be controlled with smartphones are thermostats, door locks, lighting, light switches, audio systems, window shades, wireless plugs, etc. [19].

2.1.2.2 Software

Contiki is an open source Operating System (OS) for resource constrained embedded systems and widely used in wireless sensor networks (WSNs) nodes. It features high portability, being ported to many microcontroller platforms, has small memory requirements, just a few kilobytes of Random Access Memory (RAM) and a bit higher

requirements for Read-Only Memory (ROM). The Contiki environment and its applications are developed in C, supporting an event-driven kernel with inter-process based communication structure [20]. Contiki also provides Cooja, a network simulator based on Contiki OS and emulated IoT devices, for the rapid development of sensor networks. Code developed in the Cooja simulator can be uploaded to real hardware devices, making developing and debugging sensor networks much easier [21].

The Network Simulator 3 (NS-3) [22] is also being used for IoT networks simulations. In [23], the authors implement simulations on NS-3 for optimised neighbour discovery in 6LoWPAN networks. An implementation of an IEEE 802.11ah model with sub-1Ghz PHY and MAC protocol was done on NS-3 in [24] in order to evaluate transmission range, throughput, latency and energy efficiency in dense IoT network scenarios.

TinyOS is a sensor network OS that allows application software to access hardware directly when required. Besides being micro-threaded, it also tries to assure threaded data flows among devices and to provide modularised components with little processing and storage overhead. It uses an event-based model that requires a small amount of memory and supports concurrency. Tasks are associated with events, with no blocking or polling and when CPU is idle, a sleep state is used to conserve energy [25].

LiteOS provides a Unix-based environment for WSN applications. It features a hierarchical file system and a wireless shell interface with Unix-based commands, a kernel that supports dynamic loading and multithreaded applications, online debugging, dynamic memory, file system-assisted communication stacks, software updates for kernel and user applications, bridged through a suite of system calls [25].

Riot OS is an IoT cross-platform OS for devices with a minimum of tens of KB of RAM and flash memory. It features an abstraction layer for IoT hardware and a C programming environment for development with additional limited support for C++ and STL library. Its micro-kernel supports multi-threading, priority-based scheduling, interrupt handling, and Inter-Process Communication (IPC) interface. There are available network drivers for Ethernet and IEEE 802.15.4 and for several other IoT sensors and actuators. Network protocols such as IPv6, UDP, 6LoWPAN, RPL, CoAP, etc. are also supported by Riot OS [26].

Google's Android OS for smartphones is used to control several devices in IoT networks. Google has also introduced Android Things for the development of con-

nected embedded devices through the use of Android development tools, Android framework, and Google APIs. Android Things extends the core Android framework with additional APIs for apps integration with new types of hardware not found on mobile devices. The platform is streamlined for single application use so apps are launched automatically on start-up. It is also compatible with Google's Cloud IoT Core, a service that allows connection, management, and data ingestion from millions of devices. In combination with other services on Google Cloud IoT platform, it allows collection, processing, analysis, and visualisation of IoT data in real time [27].

When IoT data is sent to cloud platforms, a framework such as Hadoop can be used to store and process data. Hadoop contains a distributed file storage system, data storage and an analytic platform and manages parallel processing, workflow and configuration administration. The Hadoop Distributed File System (HDFS) is a highly fault tolerant distributed file system, responsible for storing data on clusters. The use of the HDFS can be useful for IoT data storage as breaks data into instances and assigns the instances to cluster nodes for analysis. Data do not have to be uniform because each instance of data will be processed on different cluster nodes. HBase, a scalable, distributed database for random read/write access in Hadoop can also benefit IoT systems in relation to log analytics, capturing metrics, and also because of its integration with Hadoop, offering data consistency, failure detection and real-time queries [28].

Apple HomeKit is an IoT framework for home automation and interaction with Apple certified IoT devices, enabling and encouraging local communications. Home configuration is stored in a local database synchronised across the devices (e.g. iPhone or Apple TV) that have HomeKit apps. According to a user's permission, HomeKit apps on a device can locate home devices in the database to communicate with over the local network. It uses Apple's iCloud service for device authentication and trust management and indirectly does the replication of the home configuration database per user device. Remote access from outside the home network is possible through tunneling the messages through iCloud to a local hub in the home network [29].

2.1.3 Device Identification

Services and devices in the IoT need to be identified in order to accomplish users and other devices requests. Names help in identifying an object and its provided services,

e.g. a flood sensor, while the address identify an object in the network. The naming element, which is how a device can be identified, currently consists of technologies such as the Electronic Product Codes (EPC), usually written in RFID tags, allowing objects to be identified in a unique and global way [30], with no requirements for line-of-sight. Ucodes are part of the Ubiquitous ID (uID) architecture, which is purely network based, therefore a ucode tag simply contains a ucode, and the ucode resolution servers contain all the information about objects and places. With ID and information being separated, it is possible to retrieve and update information on an entity and retrieve information on entities related [31].

One of the most basic and cheap technologies for objects identification is the barcode, a visual representation of the Global Trade Item Numbers. Barcode readers utilise line-of-sight for reading codes at short distances, and there are still industrial IoT scenarios which require barcodes [32].

Quick response (QR) codes, a two-dimensional type of barcode, can store data in a structure that, while too complex for humans, is easily decodable by machines. QR codes are suitable for constrained devices, such as the ones found in IoT networks, and can be used for identification, login processes and for registration of new devices to a server [33].

The addressing element, which is how an object can be found in the communications network, involves protocols such as the Internet Protocol (IPv4 and IPv6). IPv6 is seen as a replacement to IPv4 due to the enormous amount of unique addresses and the simplicity of network configuration, decreasing the chances of address collisions during Duplicate Address Detection (DAD) [34], however, many devices still use IPv4 addresses.

2.1.4 Communications and Protocols

There are diverse technologies and protocols for the communications of IoT devices in a heterogeneous network. Different types of devices have different requirements regarding energy consumption, bandwidth needs, QoS thresholds, etc.

RFID is used in IoT communications as it can store sensitive data, communicate wirelessly with other objects, and identify/track objects [35]. Similarly, NFC, which like RFID allows a communication distance of a few centimetres, can be also be used with a card emulation mode on mobile devices. The NFC Data Exchange Format (NDEF), a lightweight binary standard allows data to be stored and trans-

ferred in contactless payments, videogames, and triggering wireless connections such as Bluetooth or Wi-Fi [36].

Ultra-Wideband (UWB) enable communications with low power consumption, easy circuit topology, operating in constrained conditions with high data rate transfer and the absolute signal bandwidth needs to be larger than 500 MHz [37].

Bluetooth Classic is used in IoT for device communications due to having low-power communication requirements and being available on most mobile devices; however its complex discovery mechanisms had to be addressed on Bluetooth Low Energy (BLE), which has low energy cost for device discovery and shorter delay for small data transfers. BLE's typical advertising periods are between 100ms and 1s. BLE aims to reduce cost and delay, and increase performance of discovery and it accomplishes these tasks by reserving dedicated advertising channels that are separated from connected communication [38].

Communications in the IoT also involve Long-Term Evolution (LTE) and Wi-Fi networks. Focusing on IoT, the 3rd Generation Partnership Project (3GPP) Release 8 achieved low data rates, low power consumption, and low cost for LTE Category 1, then on Release 12 LTE complexity costs were reduced by using lower key performance indicators (KPIs). Since Release 12 Q4, LTE Category M1 (LTE-M – (LTE-MTC [Machine Type Communication])) has been developed to speed the standardisation of expanding coverage, which will improve battery life and reduce complexity aiming to solve problems of short-distance wireless communications for IoT [39].

5G, the fifth generation cellular network technology, has been developed with new features dedicated to IoT, which is expected to be a major part of 5G. 5G-based IoT is a term that can be used when IoT communications are performed on top of 5G infrastructure. All existing and emerging 5G communication bands are considering 5G-based IoT features, in order to improve QoS, decrease latency, fully integrate IoT devices to the mobile network, and interconnect devices within long distances. A massive number of IoT devices with low data rate and low power consumption, known as massive IoT, is expected to become possible through the 5G-based IoT technologies. These technologies still have important open issues to address such as energy consumption, spectrum availability, concurrency and interference, network protocols, architectural design and prioritisation among regular 5G users [40]–[42].

Two important IoT technologies set to be part of 5G specifications are the Narrowband IoT (NB-IoT) and the aforementioned LTE-M. Both these technologies have been based on the LTE standard and, according to the 3GPP, will continue to evolve

with 5G, using 5G New Radio (5G NR), and fulfilling the long term 5G low-power wide-area (LPWA) requirements [43].

NB-IoT provides support for sensing and data collection applications, such as intelligent electric meters and environment supervision. It aims to satisfy the requirements of non-latency-sensitive and low-bitrate applications, therefore, time delay of uplink can be extended to more than 10s, and uplink or downlink for a single user are supported at 160 bit/s, at least. Other requirements NB-IoT addresses are coverage enhancement, with a capacity increase of 20 dB; ultra-low power consumption, with 5-Wh batteries powering one terminal for 10 years; and massive terminal access, with a single sector supporting 50,000 connections supported by a transmission bandwidth of 200 kHz [44].

LTE-M is based on the existing LTE carriers with a 1 Mbps data rate for upstream and downstream connections. LTE-M aims to provide wide coverage, massive connections, low power consumption and low module costs. Transmission gains of around 15 dB in comparison of LTE-M to other existing technologies under the 700 - 900 MHz licensed spectrum, increase LTE coverage significantly. Other benefits of LTE-M include network cell support of nearly 100,000 connections, and standby time for LTE-M terminals of up to 10 years [44].

Point-to-multipoint (PTM) communications can also benefit IoT networks, especially if it becomes a definitive part of 5G networks. 5G PTM could, for example, provide software updates to a large number of devices, which requires the network to be reliable and power efficient. Smart cities and industrial IoT applications can also benefit from PTM updates that do not affect point-to-point traffic, as these broadcast messages can be sent during off-peak time [45], [46].

Wi-Fi, which is available in different versions with different operating frequencies and throughputs, can provide very high data rates. The popular 802.11n, for example, can reach, theoretically, up to 600 Mb/s. The most recent 802.11ac, offers up to a theoretical speed of 6.9 Gb/s, which while useful for broadband access in computers, laptops, smartphones, etc., does not need to use its full potential in sensor networks. Depending on the type of Wi-Fi, devices up to a 200m distance can be covered. Focusing on IoT networks, the Wi-Fi HaLow, (IEEE 802.11ah) targets IoT applications, which usually need lower frequencies and longer ranges. IEEE 802.11ah features a transmission range up to 1 km and data rates larger than 100 kbps, bringing the classic IEEE 802.11 WLAN user experience to fixed, outdoor, point-to-multi-point applications and machine-to-machine (M2M) use cases. Wi-Fi HaLow features extended

sleep cycles and other power saving attributes [47].

WiFi-Direct, which was built based on the infrastructure mode of IEEE 802.11 and inheriting its benefits, allows a device to be an access point for nearby devices with an on-demand infrastructure for communication. As device-to-device (D2D) communication is essential for IoT, WiFi-Direct networks are very useful as one device is selected in the application layer as the group owner and becomes a central hub for all communications [48].

The contributions presented in this thesis contain experiments and simulations with devices with low and high bitrate requirements (e.g. beacons and high definition video, respectively). Therefore, the architecture and algorithms proposed were designed to be implemented on top of the network standards presented in this section. For simulation and testing purposes, WiFi and LTE were selected, due to the bitrate requirements of the rich media devices used in the scenarios presented. In 5G environments, for instance, technologies such as NB-IoT and LTE-M for constrained IoT scenarios can also avail from the benefits of the contributions of this thesis, with the adoption of schemes proposed.

There are several protocols used with the IoT technologies presented in this section to enable network communications. They can be divided in Application Protocols, Service Discovery and Infrastructure Protocols.

2.1.4.1 Application Protocols

Application layer protocols for IoT have to be designed to work on low-cost, low-power or resource constrained devices. The main IoT application protocols are presented in this section.

The Data Distribution Standard (DDS) designed by the Object Management Group (OMG) is a standard for data-centric, publish/subscribe based communication systems, with real-time and QoS capabilities. The core of DDS consists of the global data space, providing equal access to the data, with transparent horizontal, near-effortless vertical integration. The DDS API is specified in a platform independent and object-oriented manner in OMG Interface Definition Language [49].

The Constrained Application Protocol (CoAP), standardised by the Internet Engineering Task Force (IETF), is a lightweight Representational State Transfer (RESTful) protocol focused on IoT devices that need IP-based interactions using HTTP. CoAP was designed considering the limited processing and memory capacities, low data rates and high bit error rates of IoT devices. It runs on top of the UDP protocol and

also handles network congestion [50].

Advanced Message Queuing Protocol (AMQP) is a binary open standard application layer protocol for messaging-oriented middleware. The current version, AMQP 1.0, defines message format with common data types focusing on interoperability between different vendors. There are three modes of message delivery: at-most-once (each message is delivered once or never), at-least-once (each message is certain to be delivered, even if multiple times), and exactly-once (message will always certainly arrive only once) [51].

Message Queue Telemetry Transport (MQTT) is an extremely lightweight publish/subscribe messaging protocol developed by IBM for lightweight M2M communications featuring low bandwidth requirements, code footprint size, energy consumption and message data overhead, and with different QoS levels. MQTT uses TCP as the transport layer in the TCP/IP model [52]. In the publish/subscribe messaging model, there are subscribers, publishers and a broker. The subscriber receives the data that is sent by the publisher if it is subscribed to the data/topic being published, and the broker remains centred within this message exchange to manage the events. A subscriber can be subscribed to many topics at the one time [53]. MQTT topics are structured hierarchically, separated with a slash (/), e.g. house/room1/mainlight, and clients can also match multiple topics. MQTT was created to achieve three main goals: reliable degree of delivery assurance of messages, lightweight design to minimise network bandwidth and easy to implement on embedded devices with low resource requirements [54]. MQTT data is transferred as byte arrays unlike REST, which needs to define content type. There are few messages types included with MQTT such as, CONNECT, PUBLISH, SUBSCRIBE, UNSUBSCRIBE and DISCONNECT. Regarding message size, MQTT uses short headers, two bytes is the minimum packet size for a message, one byte for control field and one byte for packet length field, which is all that is needed in a DISCONNECT message. The MQTT for Sensor Networks (MQTT-SN) variation was developed for use on embedded devices on non-TCP/IP networks such as ZigBee [51].

The Extensible Messaging and Presence Protocol (XMPP), developed by the Jabber open-source community focusing originally on instant messaging (IM) applications, has in its core Extensible Markup Language (XML) streaming technology with core specification standardised by the IETF plus more than 300 extensions published through XMPP Extension Protocols in relation to publish/subscribe messaging, sensor data exchange, multi-user chatting etc. In IoT, the XMPP protocol has

received lightweight implementations for devices with constrained limitations [51].

Regarding the Hypertext Transfer Protocol (HTTP), in RESTful web interfaces for IoT environments, objects represent resources with properties. Resources can be represented by URLs and support HTTP methods (e.g. GET and POST). Therefore, applications can retrieve the status of objects via RESTful web services [55].

REST (Representational State Transfer) is an architectural style over HTTP protocol that uses a request/response model. It is stateless, which means that each request from a client to the server must contain all data necessary for the server to process the request, as the connection is always closed after the request. One of the advantages of REST is that it uses methods from the HTTP library such as GET, POST and DELETE requests and can be implemented in most programming languages and embedded devices as most operating systems include this library [56]. However, constrained devices with limited memory and power do not need all of these methods, and other HTTP features prove useless in an IoT environment. In [57], the author demonstrated how RESTful interfaces can be used to create a complex IoT structure, with several homogeneous devices. However, due to the resources required to perform HTTP requests, receive HTTP responses and, as pointed out in [58], the high usage of network resources to establish a connection each time there is any transfer of data, REST may not be the most ideal protocol for some types of lightweight M2M communications.

2.1.4.2 Service Discovery Protocols

Service discovery protocols are very important in IoT, bringing ways of locating services and resources provided by IoT devices in the network. The Domain Name System (DNS) has received extensions to better support IoT, such as mDNS and DNS-SD.

Multicast DNS (mDNS) is a service that performs the task of a DNS server without the need of a dedicated server. Clients have caches containing the pairing between names and addresses accessed through multicast queries. The targeted device transmits a multicast response and the devices that receive it store the pairing in their caches. DNS Service Discovery (DNS-SD), proposed by IETF ZeroConf working group uses the same types of queries of DNS and enables locating and publishing services in a network, e.g. printer services. After retrieving the names of the devices that provide a service, mDNS retrieves the address [59].

2.1.4.3 Infrastructure Protocols

The infrastructure protocols in the IoT can be categorised in Routing Protocols, Transport Layer Protocols, Network Layer Protocols, Link Layer Protocols and Physical/Device Layer Protocols.

Routing protocols for IoT include several protocols, such as RPL, AODV, AOMDV, OLSR, BATMAN and DSR.

The Routing Protocol for Low Power and Lossy Networks (RPL) was developed to support multipoint-to-point, point-to-point and point-to-multipoint communications. The DODAG (Destination Oriented Directed Acyclic Graph), at the core of RPL contains one root node that starts the formation of the topology by broadcasting the DIO (DODAG Information Object) messages. Then, the nodes that receive the DIO message select a preferred parent according to their rank value that considers parameters such as distance from root node, energy of link, etc. Nodes keep broadcasting DIO messages to create the tree topology [60].

The Ad-hoc On-Demand Distance Vector (AODV) [61] is a reactive routing protocol that requests routes only when necessary, keeping timer-based states at each node using routing tables. Active paths for nodes are the routes in which the node sends, accepts or passes packets. Therefore, intermediate networks are responsible for decision making in relation to packet data transmission. Topology changes are only sent to nodes in need of the information. AODV only supports symmetric node connections and routes are created by route requests and route replies. A multipath protocol based on AODV, the Ad-hoc On-demand Multipath Distance Vector (AOMDV), increases the amount of routes found route searches and discoveries. AOMDV features hop-by-hop, distance vectors, route discovery and maintenance. AOMDV is also able to correct faulty routes and find multiple paths to the destination, which can cause overhead.

The Optimised Link State Routing Protocol (OLSR) [62] is a proactive routing protocol, which maintains routing tables in all devices of the network. After all nodes update their neighbours (nodes reachable by bidirectional links) lists, Topology Control (TC) packets inform nodes of the network topology, so the routing tables can be updated. OLSR uses HELLO messages allowing nodes to find their one-hop and two-hop neighbours. Then, the sender can use the multipoint relay feature, which is based on the one-hop node offering the best routes to two-hop nodes, decreasing control traffic.

The Better Approach To Mobile Ad-hoc Networking (BATMAN) [63] is a proactive routing protocol for multi-hop ad-hoc mesh networks. Instead of calculating all routes, the BATMAN protocol maintains routing information to the closest neighbour. Nodes cannot access the full network topology, only routing decisions.

Dynamic Source Routing (DSR) [63] is a reactive routing protocol based on source routing, therefore all mobile nodes contain routing information and do not depend on routing tables at intermediate devices. Routes are found on-demand by flooding a packet with information about the source and destination node addresses, request id, and path throughout the network. Nodes, then, receive reply packets with the path information.

Transport layer protocols include the Stream Control Transport Protocol (SCTP) provides concurrent and reliable multipath data transfer via data duplication detection and retransmission. By maintaining a list of IP addresses, a SCTP association is established consisting of one primary connection and several secondary connections, but needs support from the application layer [64].

In contrast with SCTP, Multipath TCP (MPTCP) is a transport layer protocol which does not need application modifications, using traditional TCP packets. In MPTCP, data transmission is performed through multiple channel paths simultaneously, achieving a much higher throughput than a single TCP connection [65].

Network layer protocols include protocols such as LoRaWAN, 6LoWPAN, IPv4, IPv6 and Zigbee. These protocols have features and deployments for the Network Layer of IoT.

LoRaWAN, a Low Power Wide Area Network (LPWAN) specification, is the open standard communication protocol for LoRa (Long-Range), enabling a large range of M2M and IoT applications keeping low energy and infrastructure costs. LoRaWAN is the medium access control (MAC) protocol of the LoRa standard. The communication is done through a star topology with end-devices, gateways and a network server. Gateways send control signals and data messages of the end-devices to a central network server and also decode multiple signals at once. These gateways are connected to the network server using an IP connection and the end-devices connected to the gateway through a single hop wireless communication. End-devices and gateways have a bidirectional connection that supports software upgrades, over-the-air activation and multi-casting. The network server manages the network properties such as the communication data rate settings for each end-device using the adapt-

ive data rate (ADR) scheme. The LoRaWAN specification contains three classes of end-devices (A, B, and C) which differ on the timing of the receive-window [66].

IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN) allows IPv6 packets to be sent to and received from IEEE 802.15.4 based networks (low-rate wireless personal area networks – LR-WPANs). In 6LoWPAN, a LoWPAN usually has several devices sharing the same IPv6 prefix: an edge router (ER) on the border to legacy IP networks, routers (FFD – Full Function Device) working for the multi-hop packet delivery as relay points, and hosts (RFD – Reduced Function Device) [67].

ZigBee is standard of wireless personal area network focused on low-power, cost-effective, reliable, and scalable products and applications. ZigBee provides the low power wireless mesh networking and supporting up to 64,000 devices in a network with the multihop tree and mesh topologies as well as star topology. Its network layer provides dynamic network formation, addressing, routing, and network management functions. Nodes are assigned unique 16-bit short addresses dynamically and routing protocols of ZigBee allow a system or users can choose the optimal routing strategy according to the applications [68].

Link layer protocols in the IoT are mainly represented by the IEEE 802.15.4 and Ethernet protocols [69]. The IEEE 802.15.4 MAC layer, which represents the link layer contains multiple features such as allowing devices to join and leave the network, integrated Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) for secure and reliable access, time slot transmissions for real time communications and encryption algorithms. Many IoT devices have wired connections and use the Ethernet protocol. It defines packets and frames and how devices share data through the medium.

The **physical layer (PHY)** of the protocol stack in IoT needs to consider energy efficiency, spectral efficiency, cost-effectiveness, and QoS requirements.

LTE Advanced (LTE-A) was designed in order to satisfy requirements such as carrier aggregation; Downlink spatial multiplexing using up to eight-layer multiple-input multiple-output (MIMO); Downlink intracell coordinated multi-point operation transmission and reception; Uplink spatial multiplexing using four-layer MIMO; Improvements on spectral efficiency; Substantial reduction in latency. Regarding IoT, capacity and coverage enhancement can be achieved by the use of a heterogeneous network, which is a collection of low-power nodes distributed across a macrocell (homogeneous) network. Low-power nodes are considered, including microcells, pi-

cocells, femtocells, and relays [70].

EPCglobal, a GS1 initiative, consists of a collection of standard documents, the EPCglobal Architecture Framework. In its Application Level Events (ALE) middleware filters and consolidates Electronic Product Codes and related data from one or more data sources. ALE also has reading and writing functions [71].

IEEE 802.11ah (Wi-Fi HaLow) has a PHY and MAC design that operates in the sub-1 GHz license-exempt bands to provide extended range (i.e. 1 km radius) to Wi-Fi networks. It also benefits from lower energy consumption, supporting low cost battery-powered sensors operating without a power amplifier. 4, 8, and 16 MHz bandwidths are available for higher-data rate applications. Energy consumption is decreased with the use of MAC protocols with smaller frame formats, beaconless paging mode and sensor traffic priority. Power is also reduced by making the on-time for sensors shorter with the 100 Kbps minimum data rate. Thousands of nodes can be connected to one access point [72].

The physical layer of IEEE 802.15.4 is responsible for the functions of spreading, despreading, modulation and demodulation of the signal. It operates in the industrial, scientific and medical (ISM) band within three different frequency ranges. Direct sequence spread spectrum (DSSS) is used as a spreading technique, increasing the frequency of the signal in order to increase its power and reduce the influence of noise from nearby networks [73].

Z-Wave is a wireless protocol for automation apparatuses for homes and commercial environments. It can connect all major electrical devices in smart homes, up to 232 devices. Z-Wave enables reliable transmissions of short messages from the control unit to one or more devices in the network with the minimum of noise [74].

The LoRa physical-layer standard adopts the M-ary frequency shift keying (FSK) modulation and Chirp Spread Spectrum (CSS) for robustness to interference and long transmission ranges (i.e. more than 10 km). LoRa also adopts symmetric modulation for uplink and downlink, allowing terminal devices to establish D2D links and facilitates the construction of ad-hoc relay networks [75].

2.1.5 IoT Services

Ultimately, the goal of IoT devices and networks is to provide a range of services to users, businesses, industries, cities, etc. available anytime, to anyone, anywhere.

Users can get multimedia content on the Multimedia IoT (MIoT), which can be

considered as a network of interconnected objects capable to acquire multimedia contents from the real world and present information using multimedia resources. It uses multimedia objects capable to acquire multimedia contents from the physical world, being equipped with multimedia devices such as cameras and microphones [76].

The smart home concept is widely used for central controlling of lighting, heating, ventilation and air conditioning appliances and security locks. Smart home applications extensively rely on IoT WSNs [77].

The Spanish multinational Telefonica presents in [78] their platform for Smart Businesses through IoT. The platform collects and analyses the data provided by sensors, allowing users to manage real-time or historical business data in order to optimise their processes or develop new business opportunities. Device gateways, which integrate IoT protocols, understand and interpret relevant information from received data. These gateways isolate the data processing and application service layers from the device and network complexity, in terms of access, security and network protocols. A layer of data processing and storage provides intelligence over the normalised data. This layer receives and sends data in a normalised format, storing, and processing according to a set of rules preconfigured by customers. Application Services are formed by visual software building blocks to ease visualisation of this information and there are APIs for accessing IoT data, management of devices, rules engine creation and data visualisation.

A smart city is characterised by the use of information and communications technologies (ICT) to make cities more attractive and more sustainable, and unique places for innovation and entrepreneurship. It involves numerous ICT technologies, development platforms, maintenance, and sustainability, apps for evolving citizens, industrial automation, vehicular networks, disaster planning, and technical, social, as well as economic KPIs [79].

Smart grids contain household metering devices that record and send data about energy consumption to a central system for monitoring and billing of users. Metering devices provide allow an analysis of the state of the electric power grid, decreasing the number of outages, improving reliability, reducing costs for users and energy providers [80].

IoT and smart healthcare have the potential to provide benefits from drug discovery, predictive analysis of disease, epidemic early warning, preventive healthcare and monitoring, etc. Hospitals also benefit from real time information from patients and several IoT wearable devices are also used in healthcare [81].

2.1.6 IoT Security

Due to the simplicity of certain types of objects in IoT networks, security can be a very challenging area. Combined with the traditional security issues existing on the Internet, cellular networks, and WSNs, IoT also has additional challenges regarding privacy, authentication, management, and storage.

Each layer of IoT can be attacked in different ways. The sensing or objects layer is a subject of node capture attacks, malicious code injection attack, false data injection attack, side-channel attacks (SCA), eavesdropping and interference, sleep deprivation attacks and booting vulnerabilities. The network layer is target of phishing site attacks, access attacks, Distributed Denial of Service (DDoS)/Denial of Service (DoS) attacks, data transit attacks, routing attacks, unlawful attacks and other common attacks. IoT gateways can be attacked on their extra interfaces, end-to-end encryption, secure on-boarding and on firmware updates. The middleware or service management layer can suffer from flooding attacks in cloud, cloud malware injections, signature wrapping attacks, SQL injection attacks and men-in-the-middle attacks. The application layer can be target of data theft attacks, access control attacks, service interruption attacks, illegal intervention attacks, DDoS attacks, malicious code injection attacks and reprogram attacks [82].

Efforts have been made to IoT-related protocols and standards to increase their security.

The IoT-related protocol, IEEE 802.15.4, has security definitions in its link layer and MAC sublayer. It defines data encryption including the Advanced Encryption Standard (AES), and integrity verification and hop-by-hop security where every node in the communication path has to be trusted. Encryption modes are available to support confidentiality, data authenticity and integrity. Communication is protected with a single pre-shared key [83].

Regarding the network layer, a proposed combined 6LoWPAN security protocol has been proposed in [84], since the original protocol does not have security measures. The combined 6LoWPSec mainly focuses on preventing attacks on end-to-end IPv6 communications and less strongly on radio medium attacks. 6LoWPSec implements existing IEEE 802.15.4 security features, working as an extension of the MAC sublayer. The protocol deploys the AES-CCM (Counter with CBC-MAC – Cipher Block Chaining Message Authentication Code) ciphering algorithm, which provides data confidentiality, integrity, authentication, availability and malicious intrusion detection.

One of the main routing protocols used in IoT is the RPL protocol. RPL contains several security measurements, such as support for integrity and data authenticity and confidentiality with the employment of AES-CCM with 128-bit keys and RSA with SHA-256 for digital signatures. Semantic security and protection against replay attacks is also covered, with Consistency Check (CC) control messages with a counter field. A Key Identifier Mode (KIM) is used for key management indicating if a cryptographic key required must be determined implicitly or explicitly [85].

The CoAP protocol, which is one of the most implemented application layer protocols of IoT, defines bindings to DTLS (Datagram Transport-Layer Security) for secure messages. DTLS contains features for communications authentication, integrity, confidentiality and non-repudiation for application layer communications, and also addresses reliability issues from UDP in which CoAP runs on top. CoAP employs the AES-CCM algorithm for cryptography and multiple key negotiation methods [85].

Blockchain technologies can be used in order to address some security breaches in IoT. The main blockchain aspects that are relevant to IoT include decentralisation of the network, scalability, autonomy and trustworthiness. Blockchain security measures can be applied to the safe delivery, coordination and tracking of the massive amount of IoT transactions and devices. These security measures include data management at various levels, cryptography, authentication, data integrity and detailed transaction traceability [86].

After the review of classic IoT technologies, it is possible to better understand the integration of IoT with rich media features. A comprehensive survey of rich media IoT technologies will be detailed in the next section.

2.2 Rich Media IoT Technologies

IoT supports a wide range of services, from low bandwidth smart metering to high bitrate rich media applications. At the same time, there is a large number of rich media-enabled IoT devices (e.g. there are more than 0.2 million AR/VR devices globally), which generate a massive amount of data traffic at data rates, often at more than 1 Gbps and with an expected latency of less than 1ms [87]. These services use diverse solutions to enable data transmission at high quality, including network selection [88], load balancing [89], personalised content adjustment [90] and adaptive delivery [91]. More recently, there has been a significant increase in demand for very high bitrate rich media services, including VR, in general context and in particular

that of IoT [92].

This section presents existing technologies focused on the rich media aspects of IoT, as a significant part of IoT projects involve multimedia content, virtual, augmented and mixed reality, and social features.

2.2.1 Multimedia IoT Technologies

A number of multimedia technologies supporting IoT systems have been recently introduced, such as a web application framework based on the Google Web Toolkit for improving the interaction between users and IoT devices [93]. The platform contains a visualiser for operating smart objects in a graphical and functional way. Tools for managing the devices and their interactions and for web services communications were also included. The solution provides users with simple methods for integrating smart things into a flexible visualization tool, for manipulating things both graphically and functionally, and for managing things and their interactions through a thing-centric design, with a modularity manager and web service RESTful communications. It is plugin based, controlling heterogeneous devices through a UI and backend framework. Developers of the solution tested healthcare applications with the framework so the plugin running on the application server can promptly notify web clients of any emergency situation, and, in this case, may also activate alternative methods of communications such as sending an e-mail or placing automatic calls using a software PBX. The application runs in web/mobile browsers, with a visualizer. A spectrum of IoT applications have yet to be developed on the WebIoT framework in order to evaluate the technical feasibility of shipping plugins as separate pieces of the framework, so as to enable loading them at run time on the application server through the WebIoT interface itself.

The concept of Multimedia IoT (MIoT) was formalised in [76]. The authors of the concept also introduced a QoE model for MIoT, with layers regarding physical devices, network, combination, application and context. For testing purposes, the authors developed a vehicle application with remote control, and presented the best QoE scenario among different conditions in relation to bitrate, synchronism speed, visualisation speed and map synchronisation quality. Multiple scenarios considered different configurations of bitrate, map sync, speed/rpm sync and speed/rpm visualisation. Results indicated that the best QoE scenario considers accurate data as the most important factor while synchronism is the second most important.

Mass market devices that have been employed for controlling and interacting with smart homes include the voice-driven virtual assistants and smart speakers [94], such as the Google Assistant and Amazon Echo, which, however, lack the graphical capabilities of other multimedia solutions.

2.2.2 Virtual, Augmented and Mixed Reality

Virtual Reality (VR) can be defined as a way of simulating the real world by the application of the immersion theory into a virtual 3D environment, and VR devices can be considered a multimedia IoT technology [95]. In a certain virtual area, users interact with the space using their human senses in a similar way to the real world, and therefore, VR solutions mainly focus on three human senses: haptics, sight, and hearing. The main components and layers of a VR solution are the VR engine, software and database, in the system layer; input and output devices, in the middle layer; and users and tasks, in the application layer. VR can be used in many fields including military training, collaborative study rooms, healthcare, gaming, virtual tours and museums, prototyping, product modelling and viewing, etc. [96], [97].

VR platforms are becoming a mainstream technology and affordable to many users, resulting in many VR applications being developed [96]. The platforms that hold most of the market share include the HTC Vive and Oculus Rift [98]. Some VR applications are platform exclusive which is normally established through a mutually-beneficial contract between the VR platform company and the developer of the client application.

Many works have discussed the relevance of user interfaces (UI), audio and visuals for VR applications, such as the ones presented in [99]–[101]. The UI of VR systems needs to consider how users will control the applications, how space should be perceived and the integration with the entire experience. Audio is also important as it works as a guide to users and its quality impacts user experience. For example, a VR application running the reproduction of an orchestral recital should prioritise audio without compression, while a gaming application would focus on the location of the element generating a sound for user guidance. Visuals can be made of 3D polygons or omnidirectional (360°) captured images of the real world. The graphics must not have distortions and need to accurately map the area available for user movement. The processing of this type of omnidirectional visual content requires very high resolution (e.g. 4K, 8K), massive storage, and consume a large amount of

network bandwidth. Therefore, visual content delivery is a challenge and requires powerful hardware and/or compression without much loss in quality, such as the proposed compression approaches by the Joint Video Exploration Team (JVET) of ITU-T SG 16 WP3 and ISO/IEC JTC1/SC29/WG11.

VR applications are developed with specific software development kits (SDK), and there are also tools that can communicate with multiple vendor SDKs for unified development of VR applications. A popular toolkit for this purpose is called Virtual Reality Toolkit (VRTK) [102]. VRTK, however, does not support networking.

Unity, one of the major game development engines in the industry, has the capability of supporting networking in VR applications [103]. Its support for high quality 2D and 3D video games make it one of the leading development engines in VR technology.

The combined use of VR and Augmented Reality (AR) with the urban IoT data streams from different spatial locations was explored in the context of creating a Mixed Reality (MR) environment where the digital data visualisations and the physical space are mapped so that the citizens could collaborate, interact and navigate naturally [104].

The combination of AR and IoT sensors is also used to connect the virtual game content to the real world as a way to motivate people to exercise and overcome obesity [105]. These games are referred to as exergames that could potentially improve youths' health status as well as provide social and academic benefits [106].

A VR-IoT integrated layered model for the connection and interaction between people and smart devices has been proposed in [107]. A virtual platform with the locations of smart devices in a city was implemented, creating a Virtual Environment of Things (VEoT). Users are able to interact with the smart objects using VR headsets and gestures, and the platform uses a multi-protocol sensing middleware and RESTful APIs for the connection of devices and real-time communication.

2.2.3 Social IoT Networks

A new paradigm known as the Social Internet of Things (SIoT) has emerged and it incorporates social networking concepts into IoT. These concepts include a structure with guaranteed route navigation, which will ensure scalability and effective discovery of connected objects and trust between connected objects and users that are 'friends' within the SIoT platform [108].

Twitter is one of the world's biggest social media platforms, with around 500 million tweets sent each day by its about 326 million users [109]. It has an accessible API that can only be accessed with an approved developers account. Twitter has a character limit when creating tweets. This ensures that packets being received through the API will not be too large and slow down the transmission time. Twitter also prevents duplicated posts, which display an error message on the Twitter UI if attempted, in order to prevent spamming. Twitter also has been used in a number of social IoT experiments with MQTT integration [110]–[113].

IoT and rich media technologies have been integrated in order to allow user to easily access IoT data and services and a number of solutions are detailed in the next chapter.

2.3 Chapter Summary

This chapter gives an overview of main IoT technologies, and rich media technologies that converge with IoT. An overview was presented in terms of classic IoT technologies, such as sensing devices, hardware and software for IoT, device identification solutions, communications standards and protocols, and IoT services, with the main technologies implemented in this work being highlighted.

Important rich media IoT technologies regarding multimedia IoT; virtual, augmented and mixed reality; and social IoT and media networks were also detailed as they consist of an important part of the work presented in this thesis. Investigating these technologies and standards is necessary for the understanding of the foundation of the contributions presented in this thesis.

Based on the existing IoT technologies presented in this chapter, the next chapter introduces the state-of-the-art research in the areas of IoT architectures, performance-oriented schemes for IoT, mobility, and rich media IoT.

Chapter 3

Literature Review

The last chapter introduced the main IoT technologies protocols and standards used in the context of rich media IoT. This chapter presents a detailed review of important related works pertinent to the contributions of this thesis. The research topics explored include IoT architectures and performance-oriented schemes for IoT such as QoS approaches, age of information analysis, solutions for device location and relevance, device-to-device and clustering techniques, and cloud solutions for IoT. These topics are important for addressing the challenges of keeping service quality delivery in a constrained and heterogeneous network. Solutions related to mobility, which is an important IoT feature due to the variety of existing mobile IoT devices, are also presented. Rich media IoT schemes are detailed, as they allow users to interact with IoT devices and bring social capabilities to these devices. Solutions, standards and techniques related to the above topics need to be analysed so challenges and issues in rich media IoT networks can be identified and addressed.

3.1 IoT Architectural Design

Currently, while there are many efforts towards IoT standardisation, there is no single widely agreed architecture for IoT; instead there are many approaches which employ different IoT architectures.

Works such as Supervisory Control And Data Acquisition (SCADA) [114] and IETF CoRE Working Group [115], [116] try to address some issues related to interconnection or deployment of IoT in the current service and network environment. IoT deployments will involve billions of deployed nodes, sensor data with a wide range of different characteristics, real-time data streams and a large number of devices and amounts of data produced that needs to be processed or stored.

In [117], Zanella et al. discuss the design, technical solutions, guidelines and de-

ployment of a smart city through an urban IoT, on a survey of the enabling technologies, protocols, and architecture for an urban IoT. A proof of concept was done in Padova, Italy, with 300 nodes., and discusses technical solutions, guidelines, deployment of a smart city (health, waste, air quality, noise, traffic, energy, parking, lighting, automation/salubrity of public buildings), architecture (WebService, link layer, devices). However, the authors do not propose any novel protocols, using existing solutions instead.

A resource oriented architecture for the Web of Things is proposed in [118]. The authors describe the Web of Things architecture, best-practices on the RESTful principles and discuss several protocols designed to connect environmental sensor nodes and an energy monitoring system to the WWW. This work shows how web-enabled smart things can be used in lightweight ad-hoc applications called physical mashups. For this purpose, a Smart Gateway is proposed for non-Web enabled devices. Authors suggest the Web of Things is a refinement of the IoT by integrating smart things not only into the Internet, (the network), but into the Web (application layer). This work proves that the regular HTTP protocol can be used for IoT, and showed qualitatively that the integration to web browsers is one of the most prevalent benefits, and also compared non-synchronised (each request routed) and synchronised-based architectures. Although HTTP brings several benefits, optimised protocols that minimise network connection and latency remain the best option for applications where raw performance and battery life-time are critical. HTTP, as an architecture where clients initiate interactions, is not ideal for monitoring-oriented applications, which are often event-based, and thus smart things should also be able to push data to clients. Search and discovery of billions of smart things in the web is an additional challenge, in relation to locating the desired services in specific locations.

In [119] Mulligan and Olsson discuss the architectural evolution required to ensure that the rollout and deployment of smart city technologies is smooth through acknowledging and integrating the strengths of both the system architectures proposed: ICT and telecommunications. Companies like IBM and Cisco advertise smart cities and the collation of massive data sets as a way to remove humans from the decision making process, as technology would do it better. However, the paper consider it is not true for all the use cases for smart cities, proposing a novel approach on the system architecture evolution required in order deliver the innovative new “smart city” business models proposed within the city context. The authors propose an approach where mobile network architecture and ICT, or enterprise and Internet-based

system architectures are merged. They categorise business models within smart cities into environmental, economic growth, cost, safety, quality of life, connected citizens and new business models. Unfortunately, the architecture presented is high level only and the example of mobile data being used for ICT applications is too simple (phones automatically downloading e-mails, instead of the old approach of the end user actively using the phone).

A three-layer IoT architecture was proposed in [120]. These layers are: the Perception Layer, which collects information; the Network Layer, based on internet and also processes massive information received; and the Application Layer, responsible for service discovery, connecting different IoT technologies and providing various solutions. This approach, however, gives each layer many tasks and a better level of abstraction is necessary.

Considering its massive size, IoT networks need a more flexible architecture. In [121] a 5-layer model is presented. The five layers are: the Device Layer, consisting of the physical objects and sensor devices, also responsible for the identification of devices and collection of information; the Network Layer, which securely transfers the information from sensor devices to the information processing system using wired or wireless and technology such as 3G, LTE, Wi-Fi, ZigBee, etc.; and the Middleware (or Service Management) Layer, responsible for service management and has link to the database and cloud. It processes information, performs ubiquitous computation and takes decisions based on performance; the Applications Layer provides global management of the applications after information is processed in the middleware layer; the Business Layer is responsible for the management of overall IoT system generating business models, graphs, flowcharts etc., based on the data received from the Application layer, helping to determine future actions and business strategies.

The multimedia aspect in IoT was surveyed by Alvi et al. [122] under the paradigm of Internet of Multimedia Things (IoMT). The authors propose a four stage architecture for IoMT: (1) multimedia sensing, (2) reporting and addressability, (3) multimedia-aware cloud, and (4) multi-agent systems.

The five-layer approach reviewed in this section was selected for this thesis contributions for being more flexible and having a good level of abstraction. Details about how each layer is used in the proposed solutions are presented in chapter 4.

On top of the IoT architectures described in this section, several approaches have been designed for IoT. Numerous approaches related to the main topics related to this

thesis – IoT performance, mobility in IoT and rich media IoT – will be explored in the next sections.

3.2 Performance-Oriented Schemes for IoT

A number of protocol adaptations, schemes and solutions for improving delivery quality in IoT networks have been proposed. In this section, the works that have been reviewed for the design of the performance-related contributions of this thesis are described.

3.2.1 Quality of Service Approaches

QoS characteristics of IoT involve management of network resources to provide a reliable backbone to IoT connectivity. QoS approaches aim to achieve safe and predictable services, and this is possible by techniques such as managing delays, network variations, bandwidth availability and packet loss.

A recently introduced communications protocol, IEEE 802.11ah, already described in chapter 2, features a transmission range up to 1 km and data rates larger than 100 kbps, bringing the classic IEEE 802.11 WLAN user experience to fixed, outdoor, point-to-multi-point applications and M2M use cases. A feasibility study [123] evaluates the link budget of 802.11ah proves that based on a link budget assessment, the .11ah technology can be used efficiently to establish reliable communication mainly in the downlink scenario. Some link budget limitations, however, have been noticed by the authors for the uplink case. Fortunately, if more transmit power and higher antenna gains are available, the uplink case can be also successfully established.

In [124], authors propose a compact and optimised registry-based service discovery solution with context awareness for the IoT, which is more focused on constrained domains such as 6LoWPAN (IPv6 over Low Power Wireless Personal Area Networks). It uses CoAP-based RESTful web services to provide a standard interoperable interface which can be easily inter-worked with HTTP. The modular design of protocol features allows its implementation on the constrained devices. A new adaptive timer and caching techniques are introduced. The timer minimises the protocol's control overhead and energy consumption. Its grouping mechanism is based on location tags to localise status maintenance traffic and to compose and offer new group based services. The APPUB (Adaptive Piggybacked Publish) technique balances the trade-off

between service invocation delay and packet overhead by adaptively making cache available for highly requested resources. When comparing five different scenarios, with 100 and 1000 queries (Basic, With timer, With timer and grouping, with timer and APPUB, with timer, grouping and APPUB), results show a 40% decrease in invocation delay for cases 4 and 5, smaller invocation delay, 60% decrease in control packet overhead in cases 2 to 5, 90% decrease in packets received by directory agent when grouping is used (cases 3 and 5 – with APPUB, more packets are sent though). Energy consumption decreases 12% with APPUB use (cases 4 and 5). These scenarios, however, do not consider external approaches, do not experiment service composition by employing more appropriate group leaders for the groups and no experiments with multiple heterogeneous networks and physical hardware testbeds were carried out.

In [125], authors investigated the delivery of uniform connectivity and service experience to converged multi-radio heterogeneous deployments, where distributed unlicensed-band networks (e.g. WiFi) may take advantage of centralised control functions in cellular networks (e.g. LTE). Results showed that the scheme could improve the performance of a WiFi-preferred scheme based on a minimum SNR threshold. Further work includes the investigation of user equipment-based algorithms with load variation in the network.

eDOAS [126], an Energy-aware Device-Oriented Adaptive multimedia Scheme, is a scheme that uses mobile devices heterogeneity in order to provide energy-aware adaptive streaming to mobile devices within a WiFi offload scenario that also considers LTE. The authors adapt the interactive multimedia application to the underlying Wi-Fi network conditions, device characteristics and device energy consumption, in order to prolong the battery lifetime of mobile devices and maintain an acceptable user perceived quality level. The adaptation also considers values of other metrics including packet loss, throughput and Peak to Signal Noise Ratio (PSNR). When compared with the non-energy-aware DOAS, eDOAS saves at least 5% extra energy.

The offload models from eDOAS and [125] helped inspired some features of this thesis' contributions, such as considering LTE and WiFi scenarios.

A sensor cloud testbed with Adaptive QoS (AQoS) is introduced in [127]. The testbed consists of modules, which offer services that implement and study QoS models for WSNs applications based on historical data captured from a physical network. The goal of the study is to offer flexibility through reconfiguration of the cloud as a result of a historic data performance analysis. Network performance parameters

obtained are queuing delay, buffer usage, and throughput. Other variables such as packet arrival time and departure time can also be obtained through custom modifications of the simulator used. The network communication was established between one coordinator and the 15 end-devices as unicast on a single channel using Contiki OS. A polling-based network protocol was successfully implemented for the Contiki-based physical sensor cloud setup and for every sensor node, the data retrieval can be maintained at a minimal rate. Further work needs to be done to obtain incremental outcomes from the modular organization and to establish a complete testbed system.

In [128], the authors proposed a solution for service-oriented IoT, consisting of a layered QoS scheduling model with sensing, network and application layers. Metrics used in the sensing networks include information precision, energy, sensing accuracy, network life-time and overhead costs. In addition, the authors presented a framework for services, devices and network evaluation, with algorithms for decision-making in each layer. Testing demonstrated improved QoS levels and longer node life-time.

The authors in [129] tested a network protocol stack and experimental network setup, the LoRa FABIAN. QoS metrics such as Signal-to-Noise ratio (SNR), packet error rate (PER) and Received Signal Strength Indicator (RSSI) were evaluated in multiple scenarios. Results demonstrated that the location and elevation of antennas affect the quality of the network. Authors have yet to analyse combined traffic of down and uplink to determine a possible correlation when selecting optimal stations for nodes.

A congestion control scheme for IoT presented in [130] aims to achieve better delivery in applications with high packet loss. The solution, developed for the Constrained Application Protocol (CoAP), contains a back-off timer, traffic prioritisation and adaptive re-transmission times.

Authors in [131] proposed a predictive algorithm for dynamic virtual machines (VM) allocation. Authors considered a cloud IoT scenario, analysing QoS with a metascheduler architecture. QoS metrics such as costs and deadlines determine the allocation of VMs, in heterogeneous networks with low resource availability. Results indicated that the solution was able to comply with cost and time requirements. The authors, however, have yet to employ features such as other workload models, admission control and VMs migration.

A cost-effective analytical model was presented in [132]. The authors employed a finite capacity queuing system with preemptive resume service priority and push-out buffer management in the scheme. Application scenarios with low delay needs were

considered, so the model should prioritise data sensitive to delays. Testing indicated increases in performance of priority traffic and the buffering technique avoided loss of packets containing emergency data.

An access WSN gateway was proposed in [133] to provide end-to-end (E2E) connectivity for Machine Type Communication (MTC) traffic flow within the license spectrum of LTE-APro media. The solution addressed heterogeneity of applications, services and terminal devices and the related QoS issues among them. Simulation results (in services such as voice, video conferencing, FTP and HTTP) showed significant improvements in QoS performance, regarding Packet Loss Ratio delay, and Packet Delay Variation (PDV), while retaining good QoE. In Voice over LTE, however, some degradation was encountered in network QoS.

In [134], the authors described an approach to build low-rate overlay networks in LoRaWAN, ultimately finding correlation between certain QoS metrics (Received Signal Strength Indicator (RSSI), packet delay, Signal-to-Noise Ratio (SNR) and jitter) and the distance between a node and a gateway. The authors have yet to decrease packet delay times, required by certain types of overlay applications.

Quality scores are useful to simplify the complexity and diversity of applications QoSs, such as the scoring system proposed in [135] along with a mapping mechanism for multiple QoS attributes in order to find the optimal deployment decision with the highest overall application level quality score. The problem was modelled as a maximum weighted bipartite matching problem, and was solved by the use of integer linear programming (ILP). Simulations showed that the ILP algorithm outperforms the greedy matching algorithm in terms of QoS matching for applications. ILP, however, takes more time to compute results, which may not be suitable for certain IoT applications.

An approach introduced in [136] aimed to enhance energy efficiency while ensuring a desired QoS threshold by employing a game theoretical solution concept: the satisfaction equilibrium. The authors reach an equilibrium in slow and fast fading channels by using distributed schemes.

The authors in [137] proposed an algorithm for energy-centred and QoS-aware services selection. Services were preselected offering the QoS level required to satisfy the user's requirements in terms of cost, response time, reputation, reliability, and availability, then, the relative dominance relation is used to select among the preselected services the best candidate services for the composition process. The algorithm has yet to be able to handle the dynamic re-selection of services by predicting QoS and

energy parameters changes, with respect to the different composition operators used in a given plan.

3.2.1.1 Quality of Service Statistical Analysis

A number of papers have statistically analysed QoS in traditional and IoT networks. This type of analysis is useful to demonstrate the effects of algorithms on different types or devices and indicate any significant statistical differences among results.

In [138], the stochastic geometry theory is implemented to statistically analyse the random behaviour of IoT objects in femtocells and their impact on user QoS. Models generated from this analysis are used to formulate a heuristic algorithm for use in a two-level Stackelberg game. This game allows users to attempt to maximise their utilities while maintaining a balance. Such research follows a future trend towards optimising networks resource utilisation.

An analysis on QoS statistics over heterogeneous networks [139] generated important properties in terms of throughput, delay and jitter for both concise (minimum, maximum, average, median, standard deviation, and interquartile range (IQR) values) and detailed (probability density function (PDF), auto correlation function (ACF), entropy, tail analysis, and bivariate PDF) statistics. Analysis concluded that metrics gathered at the edge of the network behaved differently from the ones gathered on the backbone, highlighting the usefulness of QoS statistics in identification of network elements and in anomaly detection frameworks.

The authors in [140] collected 8363 groups of data at end-to-end wired multimedia transmission, using 4500 of these groups as training data for statistical analysis. The authors also collected 5217 groups of data at end-to-end wireless multimedia transmission, using 3000 of these groups as training data. In applying an orthonormal algorithm, flow statistics such as packet delay, jitter, packet loss and frame rate became the input vector x, while the output shows the one of four categories that the streaming is undertaking: normal transmission, a sender side CPU violation, a receiver side CPU violation and a congestion violation.

An analysis of variance (ANOVA) was taken performed in [141], and focused on QoS for IoT. The authors analysed three parameters in respect to collective QoS metrics for IoT WSNs: collective latency, collective jitter and collective packet loss. Results show that by increasing the number of nodes in networks occupying smaller areas, packet loss is reduced; however, for larger areas the packet loss increases with increased numbers of nodes.

In this thesis, ANOVA has been used in order to analyse effects of the clustering contribution on network performance because of the possibility of making multiple comparisons among the device types.

3.2.2 Age of Information Analysis

The Age of Information (AoI) has been studied in several works in areas closely related to IoT, such as mobile networks, ad-hoc networks and WSNs. The real-time requirements of many IoT services demand metrics that can be used to measure the freshness of the information being delivered to the users. AoI is the time difference between the generation and successful delivery of information to a destination, and its optimisation allow devices to update and refresh less often but still deliver information with the same quality, saving resources.

The authors in [142] proposed a solution with framing and scheduling techniques for Cognitive WSNs to optimise energy efficiency of communication systems under strict constraints on the expected AoI. Packet lengths are regularised based on sensing and communication parameters, and channel quality factors. Investigations showed that in light traffic, shorter packets are preferred to reduce waiting time for packet formation, while in heavy traffic, longer packets experience higher discard rates, increasing the service and waiting time. Packet transmission times should, then, be much smaller than the mean available interval to avoid interruptions.

A scheduling rule that can optimise the tradeoff between the AoI and energy efficiency was identified in [143]. The tradeoff was determined by the weight coefficient in the cost function. Results found helped in the identification of suitable operation parameters for the stopping rule.

An analytical model was proposed in [144] in order to evaluate the AoI of a vehicular network. Results show that the model is able to capture and approximate how the average AoI changes with respect to the beacon sending frequency, which is related to vehicle density. Higher message sending periods are beneficial for vehicles moving in sparser zones, but degrading AoI in crowded areas. Limitations in the model included inaccuracy as the mean traffic density grows.

Investigations of the problem of AoI from a network perspective were taken in [145] and [146]. The authors proposed a new technique of optimising the scheduling in the network in order to enhance the freshness of information and also developed an algorithm offers better scalability than ILP. The study has yet to be extended to

other scenarios, multi-hop networks and a setting that incorporates the freshness of information, QoS, and the fairness among the links.

In [147], the authors studied optimal ways of managing the freshness of information in communications, using AoI. An age penalty function characterised the level of dissatisfaction on data staleness and algorithms find the optimal update policy of the zero-wait policy. Zero-wait policies are not optimal if the age penalty function grows quickly with respect to the age; the packet transmission times over the channel are positively correlated over time; and the packet transmission times are highly random. The jointly optimal control of signal sampling at the source node and remote estimation at the destination were not investigated.

An experimental evaluation of AoI was presented in [148] verifying how well the queue models approximate a network for the purpose of age. Results showed that the D/D/1 queue model models the average age better than the D/M/1, specifically in low loss cases with distribution of latencies being smooth and similar. Mixing both models can provide a good prediction when latencies are less smooth.

A model for mobile caching that considers popularity and freshness of the information was proposed in [149]. The authors proved that by caching packets that have the highest request values, the number of missed requests are minimised. The evolution of the request rate was analysed in two versions of the proposed model, providing sufficient conditions for the settling time of request rate. A mechanism in which arriving packets are updated to cached content has yet to be studied.

3.2.3 Device Relevance

A metric for defining device relevance is important in IoT platforms, as it indicates whether a device is correctly placed in the network. This means that the context of the device within the network matters for performance and quality of information. For instance, smart clusters and networks can group devices according to services provided/consumed, so the right information is easily identified and quickly accessed.

The authors in [150] studied technologies with the ability to collect relevant information in real time. Real-time information in IoT, according to the authors, is crucial for leveraging the value of the IoT as data becomes intelligence, creating an intelligent environment. The development of an intelligent environment is dependent on how smart devices' connections are and how high the quality of data they produce

is. Intelligent routing protocols and new cognitive approaches must be developed to support seamless communication among IoT devices, even in a challenging heterogeneous mobile environment.

An entity selection mechanism of main features was introduced in [40], aiming to remove redundant IoT data. A feature matrix removes irrelevant features and an algorithm calculates sensor data relevance in the feature data selection process. A similarity model was used to increase efficiency and accuracy of the entity similarity search. Tests indicated that the similarity search algorithm improved the average search accuracy by around 10%, with further improvements in search speed, data transmission and storage costs.

The work presented in [151] proposed a generic framework for relevant context-aware routing, message flows and route decision methods. The messaging scheme of the framework contains route request and route reply messages, and a multi-criteria context based decision function makes the route decisions and plans alternatives for flood limitations. A reactive protocol is also proposed, adapting a generic routing scheme for sensor networks. Testing through simulation demonstrated that the approach outperformed the AODV routing protocol in the given scenario.

A passive fingerprinting identifier for IoT devices presented in [152] is able to accurately identify device types from single packets. The approach employed machine learning and genetic algorithms to automatically learn and detect unique features different IoT devices. The solution managed to classify devices at a minimum accuracy rate of 95%, and the only errors were caused by devices from the same vendor with similar network behaviours. The authors, however only consider individual packets in the algorithms and have yet to consider groups of packets for device classification.

3.2.4 Clustering in IoT

Clustering in IoT is used to increase network performance and connect devices in a smarter and more efficient way. An IoT solution for lightweight virtualisation was introduced in [153]. Applications were integrated in a distributed and virtualised environment. The authors compared two frameworks for IoT service provisioning: one based on direct interaction between two cooperating devices and one with a manager supervising the operations between cooperating devices forming a cluster. A testbed was implemented, and testing evaluated power and resource consumption. Tests indicated that lightweight virtualisation allows for the execution of a wide range of IoT

applications, while enabling the desired abstraction level with advantages in terms of manageability and scalability.

The authors presented in [154] a matching-value based IoT service composition scheme. The work focused on the representation, discovery, detection, and composition of services. The authors also introduced an algorithm for cluster-based distribution, allowing a more robust and trustworthy decision process, using a distributed consensus method in order to improve robustness and trustiness of the decision process. The scheme, however, did not considered all phases of the services life cycle.

A Multiple Base station and Packet Priority-based Clustering scheme (MBPP) for IoT to evaluate performance through computer simulations was proposed in [155]. The proposed protocol sends sensor data based on priority (real-time data, emergency data e.g. forest fire). Clusters are formed according to the amount of devices, and guided by a cluster head that select active IoT devices, providing network coverage of the cluster. The devices store the sensed data in memory and schedule to transmit to its cluster head on its allocated timeslot based on data subscription and priority. Data is aggregated from all devices and sent to a base station that transmits the data to a central server through Internet. The number of stations vary according to network size. When compared to two other approaches, energy consumption is decreased by 90% and 75% less transmissions happen. However, this work has some limited assumptions: the network comprises a large number of sensor-based IoT devices; all devices have the same initial energy and they know their locations; it uses time division multiple access (TDMA) scheme with variable length timeslots; and real-time and emergency data will have the highest priority, but lower priority data can preempt real-time if they cannot transmit for a long period of time for the continuous arrival of higher priority data.

An architecture for deploying WSNs, with several sink nodes and layers, was proposed in [156]. The architecture considers IoT-cloud integration and also included a routing protocol and an algorithm for selecting cluster heads in multi-layer multi-sink clusters. The algorithm performs load balancing while considering load fairness and energy. Results indicated improvements in network life time, even though the solution does not include peer-to-peer (P2P) links.

A clustering algorithm based on load adjustment for IoT in an SDN architecture was presented in [157]. Data-centres and storage units are centralised by an SDN controller, which calculates load-balance, communication cost and energy in a clustering algorithm. The authors showed that the proposed algorithm significantly extends

battery life of IoT devices and the amount of data sent increases.

A general framework for quick decision making in a rapidly changing IoT environment with a Service-Oriented Architecture (SOA) was presented in [158]. Data from clusters were merged to provide a global consensus on the final outcome. For the proposed implementation, real time data from pressure sensor and temperature sensors were captured with the help of a Hyper Terminal that provided a real time data feed as the values were being read. Unfortunately, the solution only clustered two types of sensor data, so further testing is required.

In [159], the author proposed a mechanism for context-aware communication across IoT networks, fulfilling IoT specific communication requirements, while keeping the communication overheads to a minimum. It is also adaptive for different types of IoT networks, available communication interfaces, subscriber-entitlements and changing communication network conditions. An Interconnection Gateway (ICG) was proposed and it is able to understand and maintain relevant aspects of IoT function-specific communication requirements; determine suitable modes of communication based on IoT subscriber's entitlement and fulfil the communication need as appropriate. Further testing needs to be done on different environments as the authors only tested a vehicular network.

The authors of [160] proposed a scheme for a multi-channel cluster-based framework to provide reliability for mobile Wireless Mesh Networks. The solution considers mobility of mesh points that degrades reliability of network. The novel cluster-based framework allows the network to be managed more efficiently with a cluster head, cluster members and cluster gateways. Every cluster uses a different channel for intra-cluster communication, avoiding collisions. Difficulties in inter-cluster communication due to high mobility between two clusters are addressed by using a special channel. The limitations in the solution consists assuming that all mesh points have the same specifications and their times are synchronised.

A multi-objective optimisation problem of the tradeoff between QoS provisioning and energy efficiency was addressed with a non-dominated sorting genetic algorithm in [161], with the authors also introducing a method to select optimum cooperative coalition for cluster heads and cooperative nodes, by using exhaustive search combined with quantum-inspired particle swarm optimisation.

The Equalised Cluster Head Election Routing Protocol (ECHERP) presented in [162], focused on energy conservation through balanced clustering. The network is modelled as a linear system and using a Gaussian elimination algorithm, combina-

tions of nodes that can be cluster heads are calculated to extend network lifetime. Performance evaluation evidenced the effectiveness of the protocol in terms of network energy efficiency when compared against other protocols. ECHERP has yet to be enhanced by aggregating other QoS metrics and time constraints.

An adaptive method was developed for determining the number of clusters that can be found in a stream based on the data distribution. An online clustering mechanism was used to cluster the incoming data from the streams. A set of experiments in a traffic use-case scenario, compared against a non-adaptive stream cluster algorithm, showed significant improvements in the cluster quality metric of silhouette coefficient [163].

The clustering solutions presented in this section were useful for the development of the clustering techniques implemented in this thesis, as they were proven to be useful in optimising resources in the network, while prioritising network performance. In Figure 3.1, an example of device clustering is presented, with objects O_{31} and O_{14} being re-attached to other smart gateways and receiving new IDs (O_{14} and O_{24}, respectively).

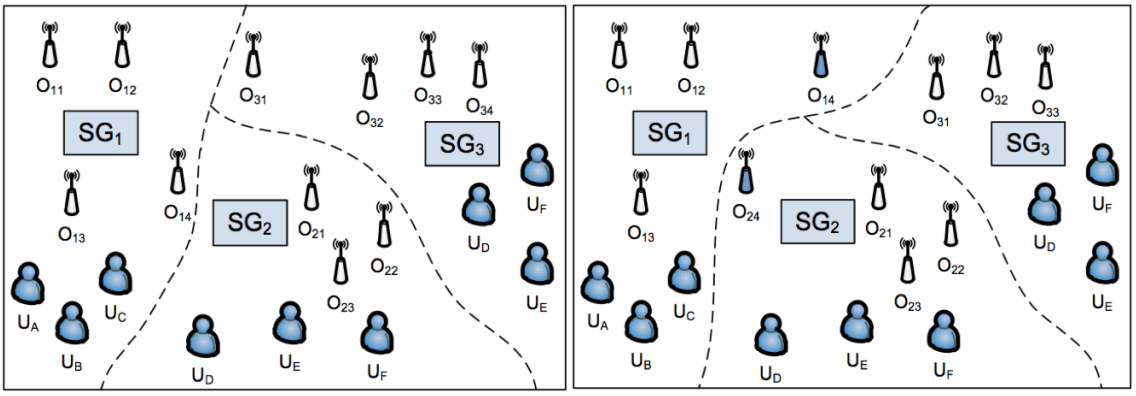


Figure 3.1: IoT Device Clustering

3.2.5 Device-to-Device IoT Communications

Technologies that are relay-based and allow devices to intercommunicate directly, such as WiFi Direct, have allowed Device-to-device (D2D) communications to become a reality. D2D communications are essential for IoT heterogeneous device interconnection, especially in scenarios where gateways or base stations/antennas are not available.

A modular architecture for IoT with D2D features was proposed in [164]. The architecture contains a direct D2D communication sub-layer. The authors use D2D to address the needs of the next generation IoT applications, such as low latency, robustness of connection, support of different data types, reconfigurability and wide coverage.

A hybrid resource allocation algorithm for D2D communications was proposed in [165]. A centralised interference mitigation algorithm and a distributed joint channel selection and power allocation algorithm were proposed in order to achieve a near-zero infeasibility ratio of the spectrum capacity due to interference, while improving energy efficiency.

The authors in [166] introduced an overlay-based relay assisted D2D communications IoT architecture. The solution improves cellular networks, and by adopting low power pico base stations for relay, it facilitates D2D communications. The results showed that the utilisation of green energy was maximized by balancing the energy among the relay base stations.

3.3 Mobility Approaches for IoT

Several IoT devices are mobile, which brings several challenges for researchers. Therefore, a number of solutions have been proposed in the field of mobility for IoT.

A framework for path planning in emergencies was proposed in [167]. The framework considers mobile IoT devices and indoor human movement, in order to achieve optimal evacuation times. Results demonstrated that the prototype performs better than other existing solutions.

The authors in [168] introduced an agile data delivery framework for smart cities. The framework contains multimedia applications for vehicular networks (VANETs) and WSNs. The solution aims to select optimised paths for packets while maintaining high QoS levels. Metrics studied include energy, delay and throughput, and results indicated less overall delay and longer network life time. The effects of the framework in a larger network have yet to be studied.

The authors proposed a social IoT (SIoT) architecture for handoff between mobile and fixed access points in [169]. Android devices can share, download and use geo-information such as travel information of scenic spots, landmarks, and other points of interest during group tours. Tests indicated lower energy consumption, service time and cost.

A study on a routing-based mobility management architecture for IoT Devices was presented in [170]. The scalable routing architecture uses Bloom Filters, a space-efficient randomised data structure for membership queries with false positive errors. The solution for IoT applications was implemented in an environment wherein 500,000 IoT devices move in a square field independently. The authors introduced an aggregation scheme that reduces the size of routing information; however, the scheme sacrifices accuracy and the solution does not suppress overheads to update routing information.

In [171], the authors presented a protocol to carry out inter-WSN mobility inside of an architecture that had been defined at a hospital. This protocol among other tasks, carried out the moving signalling, decreasing the number of interchanged messages among the mobile IoT nodes. An architecture was developed to offer flexibility in communications, monitoring and control. The architecture included communication technologies such as 6LoWPAN and RFID/NFC. The authors, however, did not analyse if power consumption of the 6LoWPAN sensors was affected with the introduction of the mobility protocols.

MOSDEN, Mobile Sensor Data Processing Engine, was introduced in [172]. MOSDEN is capable of collecting data from multiple different sensors and process them together and is also compatible with the Global Sensor Network Middleware that runs on the cloud. The engine performs data processing and analytics before transmission over a network and processes sensor data based on SQL-like queries. The solution was based on a plugin architecture, where each plugin translates generic communication messages to sensor specific commands in order to enable communication between MOSDEN and a specific sensor. The middleware was written in Java and runs on Android based devices. Three of these devices with different resource limitations were used in the simulations. The authors have yet to develop a configuration model that can be used with different types of devices.

3.4 Rich Media IoT Schemes

Rich media IoT could have a significant impact in many areas such as healthcare, education, intelligent transportation, agriculture, etc. The integration of IoT with multimedia technologies, VR and social networks, will introduce a major shift in the paradigm, as surrounding objects and things will start being connected as well as social.

With rich media technologies being integrated to IoT, users will be able to seamlessly control their devices through multiple platforms, and IoT devices' feedback can be received in several types of screens, headsets, smart speakers, etc.

This section will present and review some works in the state-of-the-art of rich media and IoT integrated solutions, so that challenges and new areas of research can be identified.

3.4.1 Multimedia IoT Solutions

The integration of multimedia communication and IoT can introduce a dramatic change in the way people are living. For example, in case of disaster management the first responders could get real-time multimedia streaming data of the incident before they actually reach the scene, so they can anticipate the scale and emergency of the disaster.

However, it is known that multimedia traffic is power and bandwidth-hungry and has strict QoS requirements. Thus, the technical challenges posed by transmitting real-time rich media data within IoT need to be identified and addressed.

In [173], authors proposed a home automation system based on a 3D virtual world. The goal is to help users interact with smart homes in a friendly and easy-to-use environment. A computer runs the graphical user interface, which is controlled with a mouse. The MPEG-V standard is used along with a user-defined protocol. The authors did not consider VR for user interaction and did not implement different protocols in the solution to analyse the performance of the platform.

The authors in [174] introduced a web-based interactive framework for visualisation and authoring of indoor IoT environments containing sensors and devices in homes or small offices. An indoor 3D scene was configured to display attributes of the available physical sensors and actuators, also allowing the visual programming of virtual sensors and actuators. The testing of the proposed framework, however, did not consider a synchronisation scheme and did not include real physical sensors and actuators, which were emulated by the server.

In [175] the authors proposed an IoT approach that could improve education and streamline organisations. The aim was to use IoT technologies to improve multiparty audio-visual collaboration during online classes. Factors such as noise caused by open windows, poor lighting or high volume of speakers can cause disturbance to the overall collaboration in the online sessions. In order to improve these conditions,

sensors could be installed in the classrooms analysing the ambient conditions and report if the room is ready for remote collaboration. The continuous monitoring can send notifications or take some actions automatically (e.g. decrease the volume of incoming audio or adjusting the lights). Automatic detection of voice activity could unmute microphones improving the quality of communications during an online class. During online sessions, an accurate 3D location system (e.g. Ultra-WideBand), based on tags location, could improve zooming and position of cameras. Even though the ideas were very interesting, the authors did not implement and test them.

A multicast routing solution for supporting multimedia communications in IoT was introduced in [176] with fast multi-constrained multicast routing algorithms. Point-to-multipoint trees were created for meeting multiple QoS constraints required by real-time multimedia communications. Authors set a network size in the experiment from 100 nodes to 500 nodes, and the number of destinations is set to 5 and 10. Analytical and experimental results demonstrated that the proposed algorithm was superior to a representative multi-constrained multicast routing algorithm in terms of both speed and accuracy. Algorithms proposed by the authors, however, assumed the availability of current network topology and the QoS constraint information associated with network topology.

A novel data model and storage infrastructure for the IoT was introduced in [177], using a document-oriented approach to support both heterogeneous and multimedia data. The solution was built on top of the CouchDB NoSQL database server and offers a RESTful API that provided a set of features to IoT applications, such as replication for load balancing, distributed query processing, and notifications. Proposed optimisations reduced the transfer time of documents containing multimedia data; however, the solution was not a complete framework for web-of-things yet, as authors still wanted to add support for online social networks.

The approaches reviewed in this section were used as inspiration for the development of the multimedia elements of this thesis, especially in relation to protocols, user experience and application scenarios. The need for user-friendly tools for IoT devices is evidenced by the research efforts in the creation of multimedia solutions to control these devices, especially in smart homes.

3.4.2 VR-IoT Integration

VR and IoT have been recently integrated in order to maximise the high interoperability of IoT services with the intuitive nature of VR. This integration also allowed the creation of an immersive virtual representation of a smart cities, offices and homes. These VR-IoT integrated platforms can provide a customisable and user-friendly environment for controlling real IoT devices in virtual environments.

The creation, implementation, and evaluation of a Unity game on an engine-based application layer for a brain-computer interface was proposed in [178]. The game, which integrates 3D graphics, VR, and IoT devices, was only tested in a local-network environment.

An end-to-end system architecture that underlies IoT networks for seamless VR space was proposed in [179]. The architecture contains five stages: acquisition, classification, virtual image and video reconstruction, transmission, and consumer processing. The authors, however, did not present a testbed to demonstrate the architecture.

A VR engine for immersive smart city visualisation was proposed in [180]. The engine contains network features for big spacial data and online sharing with hash-based P2P capabilities. Users can navigate in a map with real geographic space using avatars, and performance evaluation tests demonstrated high usability and user satisfaction. The engine, however, does not allow users to interact with smart devices of the city, being more useful for city planning purposes.

A platform for hyper-connected IoT-VR was presented in [181]. Users and devices can be interconnected in a virtual space, which contains customisable remote services. The authors built a testbed, which included a cloud server, but did not analyse network latency and performance using different protocols, and did not implement a synchronisation scheme to guarantee consistency between real and virtual devices.

The authors in [182] introduced a VR approach for exploring software-built cities using VR headsets and interactions with gestures. The authors also created a reusable gesture model for operating 3D environments. In order to test the solution, participants of the tests were asked to rate the usability of gestures and the overall VR experience. Results demonstrated that participants praised most of the introduced gestures.

Regarding VR synchronisation, the authors in [183] implemented a computing offloading scheme via an mmWave 802.11ad 60GHz wireless link in order to share 360

VR content with a PC. Synchronisation algorithms were employed at packet level,

handling real-time video transmission without packet loss.

Based on the idea of creating innovative ways of controlling IoT devices, VR seems like a suitable approach due to the ever-increasing consumer adoption of VR technology. The works reviewed in this section demonstrate the importance of a pleasant and user-friendly VR experience, and the feasibility of its use in diverse fields.

However, the approaches for integrating VR and IoT presented in this section have limitations in their implementations, not showcasing the full potential for a VR-IoT solution. Some of the approaches do not have cloud capabilities; do not fully implement user interaction controls for IoT devices; are only applicable for certain scenarios; do not guarantee consistency between real and virtual devices; or do not test a variety of protocols.

Therefore, there is a need for an integrated solution that allows end-users to interact with IoT devices using intuitive controls, guarantee that real and virtual devices are synchronised, and test the performance of different communication protocols used for the VR and IoT integration. Another feature possible with a VR-IoT system is the ability of controlling objects remotely, something that can be addressed by hosting certain functionalities in cloud-based servers, therefore, the next section focuses on cloud solutions.

3.4.3 Cloud-Based IoT and VR Solutions

There are a number of works exploring IoT and VR capabilities in the cloud, relieving local resources and off-loading tasks to cloud-servers.

In [184], the authors explained that the virtualisation of IoT devices allows users to easily share the devices. They also presented the multitenancy feature of cloud computing as an enabler for resource sharing to multiple users over spatial and time distributions.

The authors in [185] presented a cloud-based smart home platform for integrating smart home IoT services and managing operations. Users can manipulate and visualise the devices in the platform with a user interface. The authors also compare the data traffic and latency generated by HTTP and MQTT protocols.

An approach on sharing IoT devices among end-users was presented in [186]. The service-oriented approach allows IoT devices to provide data and resources, which are available in a cloud platform, at any given time from any place with connectivity. The approach gives the opportunity for application developers to reuse information

provided by the users and create applications on top of the solution.

The authors in [187] proposed the concept of local IoT automation clouds. The solution design was demonstrated in a compartment climate control. Automation and real-time performance were also analysed.

VR has also benefited from cloud technologies. Computational off-loading, image processing and mobile capabilities are among the features found in cloud-based VR solutions. The authors in [188] investigated how extensive rendering being off-loaded to a cloud/edge/local server can make the VR/AR processing to be lighter on the headsets. There are, however, challenges from bitrate to latency requirements, and possible solutions to enable cloud/edge-based wireless VR/AR include asymmetric video encoding and rendering, and hybrid-casting.

In [189], the authors presented a solution which uses off-loading as a service, in order to to resolve the mismatch between how mobile devices demand computing resources and how cloud providers offer them. They also designed a VR framework able to seamlessly operate a 3D game by computation off-loading.

The works reviewed in this section demonstrate that cloud technologies have been used in IoT and VR, mainly for data virtualisation and off-loading. The benefit of accessing data from anywhere brings challenges such as latency in communications, which can be an issue especially in VR applications. Therefore, there is a need for performance analysis of local-based and cloud-based approaches, and testing of the feasibility of having features in the cloud.

3.4.4 Social IoT Platforms

Finalising the review of rich media IoT works, some social IoT platforms will be described, as IoT networks have recently adopted social features, which are going to be useful for the contributions of this thesis.

In [190], the authors stated that the majority of young people receive information on world events through the use of social media platforms such as Twitter, Instagram, YouTube, Snapchat and WhatsApp, to name a few. Another application of social media analytics was presented in [191], aiming to use information concerning the emotional states of users through content they shared. The real time sharing of these updates could create faster emergency responses or improve crime control. Social media allows users to update a status which can be viewed by the user's friends. Similarly, with the concept SIoT, an object can update its status and location, and be

seen by the friends of this device, which may be people or other objects.

An implementation of an SIoT network in a smart home environment using Facebook was introduced in [192]. This study presented the benefits of using social media features in IoT to monitor and control connected devices in a home environment, taking advantage of the Facebook security system to keep data private and visible to a chosen audience. The authors, however, did not perform any network analysis.

Another example of a social IoT system is represented by Waze [193]. Waze is a real-time community-based traffic and navigation application that enables users to update traffic-related information, such as: road conditions, traffic intensity, accidents, etc. This information is then used to find the best route to get to a destination. This represents a good example of how the connected cars could take advantage of the collective intelligence of the community of connected cars.

Social IoT is envisioned to be applied to other areas as well, such as connecting medical devices and people by forming a social network of doctors, patients and family members [194]. Or the connected shoes provided by Nike [195] so that the customers could compete against each other in various sport activities.

3.5 Rich Media IoT Application Use-Cases

For completeness, this section aims to review a number of application use-cases that involve IoT and several rich media types of content. These use-cases are classified in five categories: Smart Surveillance Systems, Autonomous Vehicles, Healthcare, eLearning and Smart Agriculture.

3.5.1 Smart Surveillance Systems

One of the most common and important rich media-based IoT applications is represented by smart surveillance systems, which are mainly used for real-time road traffic monitoring, security control, smart parking, etc. These smart surveillance systems, consist of cameras or imaging sensors installed in key locations depending on a specific purpose, such as smart transportation and urban surveillance applications, smart homes, smart retail, etc. as indicated in Figure 3.2.

The data from the surveillance systems is sent to a Control Center (e.g., cloud) where is stored and analyzed. Depending on the application requirements, the data at the Control Center is processed using some key techniques, such as: crowd detec-

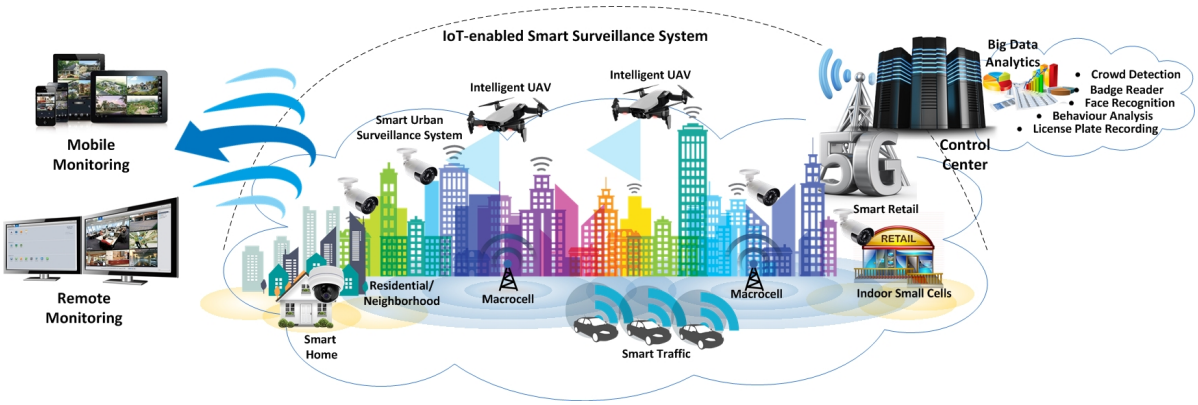


Figure 3.2: Smart Surveillance Example

tion, badge reader, face recognition, behavior analysis, license plate recording, etc. In general, these techniques require high-complexity image/data processing technologies and algorithms that need to provide accurate and reliable results. For example, in case of license plate recognition several methods have been adopted in the literature, such as: image processing [196], profile-based filter [197], contrast between grayscale values [198], morphology-based solutions [199], [200] color-based methods [201], unified method of automated object detection [202] or more recently, convolutional neural networks [203], [204].

In the context of the smart home [205] a smart surveillance system will monitor the house and evaluate through data processing is an intruder is detected of not. Upon the detection of an intruder, the house alarm will be activated as well as a notification will be sent to the user's mobile device.

The smart surveillance systems could also enable the implementation of highly accurate video-based indoor positioning systems [206]. These systems could be used for improving the security in smart retail by providing real-time and accurate indoor positioning of the customers.

These smart surveillance systems represent key enablers for smart cities aiming at improving the quality of life for the citizens. One interesting use case scenario is described in [207], where the authors tackle the issues of safety at the beach side. In this context, smart surveillance cameras have been deployed and the video-based surveillance is used for downing prevention by detecting and monitoring the human activity in the water.

Additionally, the integration of intelligent unmanned aerial vehicles (UAV) within the smart surveillance systems has been proposed in [208]. In this context, the police

could deploy patrol UAVs to act as mobile surveillance platforms where the fixed CCTV cameras are not in place or the target is on the move. This solution could increase the security and reduce crime related costs.

However, the integration of these solutions within the concept of smart cities will require a number of IoT devices greater than 1000-1million per city, generating a massive amount of video-based data traffic at an approximate capacity of more than 10-100 million GB of real-time data per city per day with an expected latency of less than 1ms [87]. This represents a great challenge even for the highly anticipated 5G networks.

3.5.2 Autonomous Vehicles

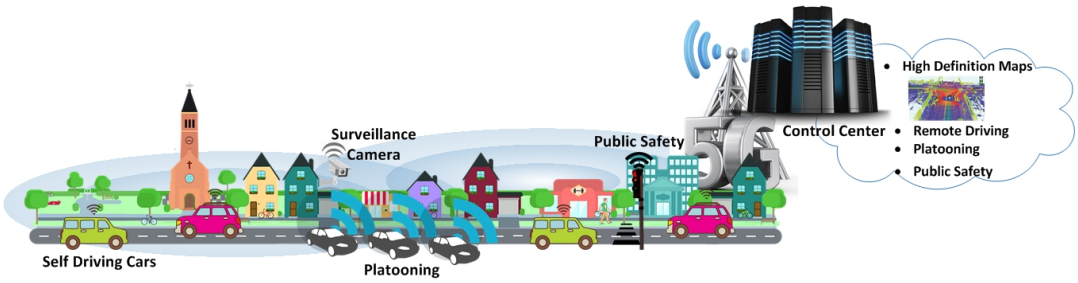


Figure 3.3: Autonomous Vehicles Example

One exciting vision for the future 5G networks is the autonomous vehicles, where most of the cars on the road will be connected and controlled through IoT. This will facilitate information collection, processing and analysis that could be used to improve transportation efficiency, ensure traffic safety and reduce traffic load. Within industry, there are currently a number of companies already working on driverless cars, such as: Tesla, Uber, Toyota, Hyundai, Waymo, BMW, Ford, etc.

As illustrated in Figure 3.3, the IoT data collected from cars, pedestrians, road side infrastructure, etc. could be sent to a Control Center for storage, processing and analysis. However, to enable fully automated vehicles, a highly accurate and performing intelligent control system is required to detect and avoid obstacles, pedestrians, cyclists and other road users. This is achieved by the wide range of IoT objects and sensor technologies deployed as well as the Vehicle-to-Everything (V2X) communication.

One of the core technologies for self-driving vehicles is the use of High Definition (HD) maps [209]. The HD maps give highly accurate precision at centimeter level which is very important especially for driving in crowded urban areas. The HD maps are created by combining the information received from the IoT objects and LiDAR (Light Detection and Raging). LiDAR works by sending and tracking down light pulses in order to identify the surrounding objects. The benefits of using LiDAR is that it overcomes the drawbacks of Radar (radio waves) for short range and Sonar (sound waves) for long range. Consequently, LiDAR can accurately identify and map the near and far objects/obstacles on the road. Additionally, in order to improve the efficiency and quality of the HD Maps, machine learning could be integrated [210] to create a self-learning platform that continuously analyzes the data on the fly and introduces safety improvements.

To enable autonomous driving, the number of IoT devices required per vehicle is somewhere around 50-200 objects generating a massive amount of real-time data including rich media with a capacity of more than 100 GB per vehicle per day and with an expected latency of less than 1ms [87].

3.5.3 Healthcare

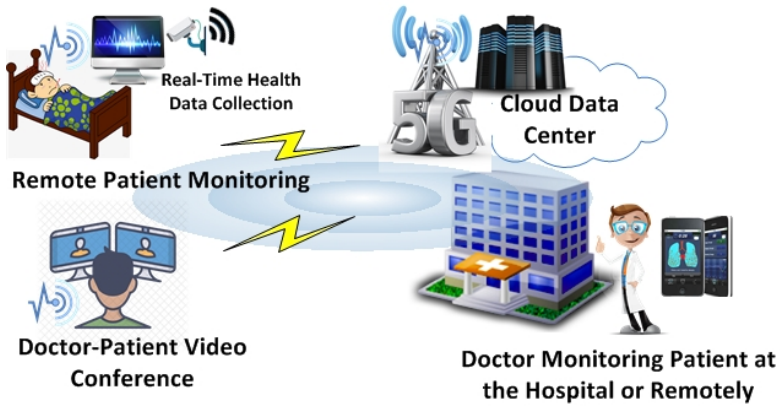


Figure 3.4: Smart Healthcare Example

The penetration of IoT-based solution into healthcare led to the appearance of Internet of Health Things (IoHT) [211]. IoHT enables the connectivity between the patient and the healthcare facilities. The use of biomedial sensors help the individual to monitor any of the vital signal of the body, including heart rate, blood pressure, body temperature, oxygen in blood, airflow, etc. As illustrated in Figure 3.4, the data from

these IoT objects could be collected, stored and analyzed in the cloud [212]. The data could be accessed by the doctor either in the Hospital or remotely in real time. Additionally, the health data analysis could provide suggestions for health improvements to the patients through their mobile devices.

Real time video streaming for remote patient monitoring and real-time statistics observation could be combined with the data from other IoT objects to help doctors in providing more accurate diagnosis and advice for their patients. Moreover, there is also the possibility to enable real-time video conference between doctors and patients regardless of their location. In this way, the hospital costs and clinical time are reduced while the quality of care is increased.

The integration of IoT within ambient assisted living has seen great potential in offering support especially for elderly people enabling them to live a more independent but safe lifestyle [213]. By using wearable and ambient IoT devices and sensors, one could monitor the daily living activities of the ageing population, their social interactions and take control over their chronic disease management in order to improve their well-being.

The continuous advances in technology led to the development of remotely controlled robotic surgeries or computer-assisted surgeries where machines carry out surgical procedures. It is expected that by 2020 the surgical robotics market to double reaching to $6bn[1].

In case of remote surgery, the number of IoT devices required is somewhere more than 10-100 per surgery, generating real-time streaming data with an expected latency of less than 200ms [87].

3.5.4 eLearning

Until recently, the concept of smart classroom/school referred to the classrooms that had integrated one type of smart device, like the smart board. However, nowadays the concept of smart school makes use of IoT as a computational system that collects student-related data from the surrounding smart devices (e.g., eBooks, tablets, various sensors, smart displays, etc.) to provide a personalized student learning experience as well as enabling remote or on the move eLearning as depicted in Figure 3.5. By creating this multimedia-based IoT-centric student learning environment enables flexibility for teachers and students enabling them to access the learning content

[1] https://www.telegraph.co.uk/business/2017/11/28/robotic-revolution-robot-could-next-surgeon/

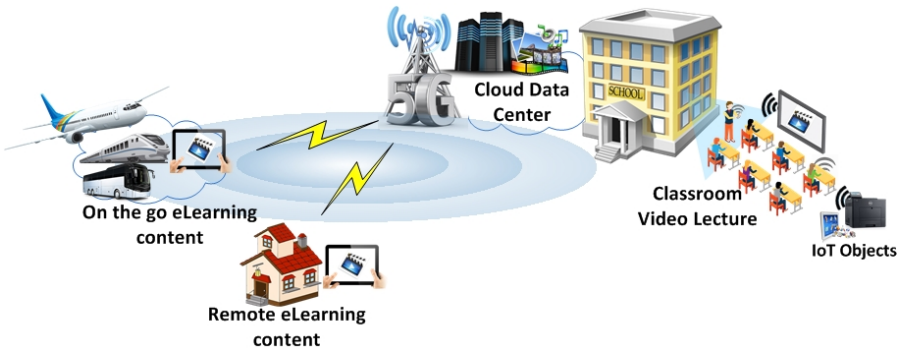


Figure 3.5: eLearning Example

from anywhere at anytime and from any device [214].

Moreover, the educational system could avail of IoT together with AR technology to provide a more relevant, engaging and interactive courseware to the students to keep them motivated and interested in the subject [215]. AR gaming enables the delivery of the learning content through mobile application games that facilitate the learning as well as the knowledge evaluation through playing.

3.5.5 Smart Agriculture

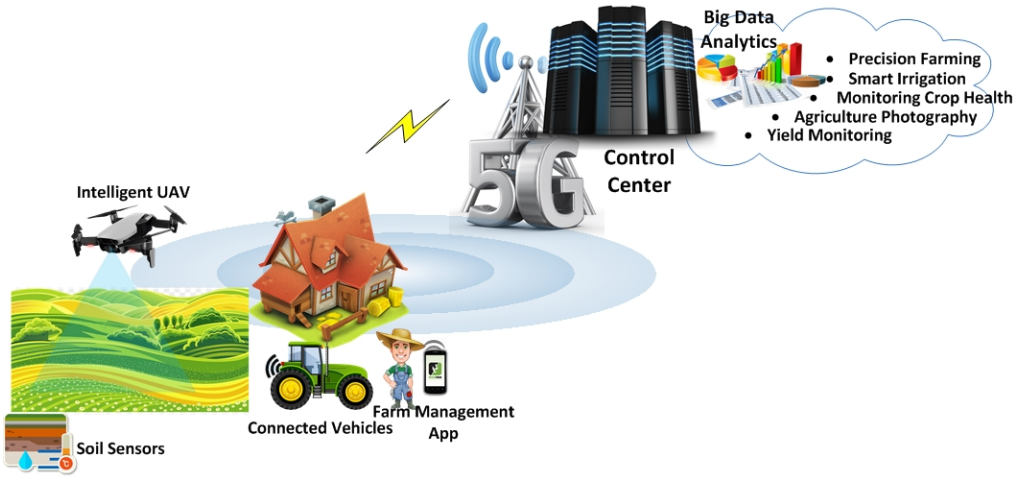


Figure 3.6: Smart Agriculture Example

The use of IoT in Agriculture has seen great potential especially for precision agriculture representing an useful asset for farmers. In this context, the IoT devices could provide precise information about the production and different environmental factors: soil quality, irrigation levels, pest and fertilizer [216] or green house produc-

tion environment [217]. To enable the smart agriculture environment the IoT sensors could be deployed on tractors, in the field and soil or intelligent UAVs could be used to collect video-based information from above as illustrated in Figure 3.6. All this information collected from the IoT sensors could be sent to a central cloud for storage, processing and analysis.

Various information management systems approaches could be used to aid farmers in analysing and interpreting the data in order to optimize the growing environment and promote smart cultivation at any point within the production chain [218]. Consequently, integrating IoT within agriculture could enhance productivity and also enable opportunities for automating farming methods.

3.6 Chapter Summary

This chapter aims to present a survey of the literature, state-of-the-art solutions, and use-cases in the areas of IoT architectures, performance-oriented schemes for IoT, mobility, and rich media IoT. Solutions, standards and techniques related to these topics need to be analysed so challenges and issues in rich media IoT networks can be identified and addressed.

A thorough review of related works was also presented, and IoT solutions were discussed in the context of this thesis, highlighting their gaps, some of which addressed by the contributions of this thesis. Some of the related works do not perform analysis of network performance, other research outputs do not discuss comparatively different protocols, while other works do not consider synchronisation in order to guarantee consistency between real and virtual devices.

The understanding of the approaches presented in this chapter was important for the development of the solutions presented in this thesis, which aim to improve the QoS of IoT networks and improve user experience when operating IoT devices. In the next chapter, the architecture and design that enables the development of such solutions is presented.

Chapter 4

Solution Architecture

This chapter presents the solution architecture that supports the contributions presented on this thesis, aiming to address challenges raised in the last chapter, after the review of state-of-the-art research. The layers, major blocks and components of the rich media IoT architecture are presented in this chapter, with a brief introduction of the solutions that will be presented with more details in the following chapters.

4.1 Proposed Architecture

The proposed architecture is composed of clustered IoT objects (things), smart gateways, the IoT Integration Platform (ITINP) aiming to improve clusters and overall network performance, and the VR IoT Platform (VRITIP) aiming to improve user experience when operating the devices connected to the network.

The major blocks and components are illustrated in Figure 4.1 and are described in the next sections.

4.1.1 Layered Architecture

Each component (i.e. objects, smart gateways, VRITIP and ITINP - which will be described in the next sections) is deployed at a different layer in the proposed IoT layered architecture, presented in Figure 4.1. The contributions of this thesis consider a five-layer IoT architecture, as discussed in [121], [219].

In this layered architecture, the Objects layer comprises the devices, sensors, etc. and communicates with the Network layer (sometimes referred to as abstraction layer), where the smart gateways are located. The smart gateways are responsible for clustering local devices, gathering and sending data to improve performance and

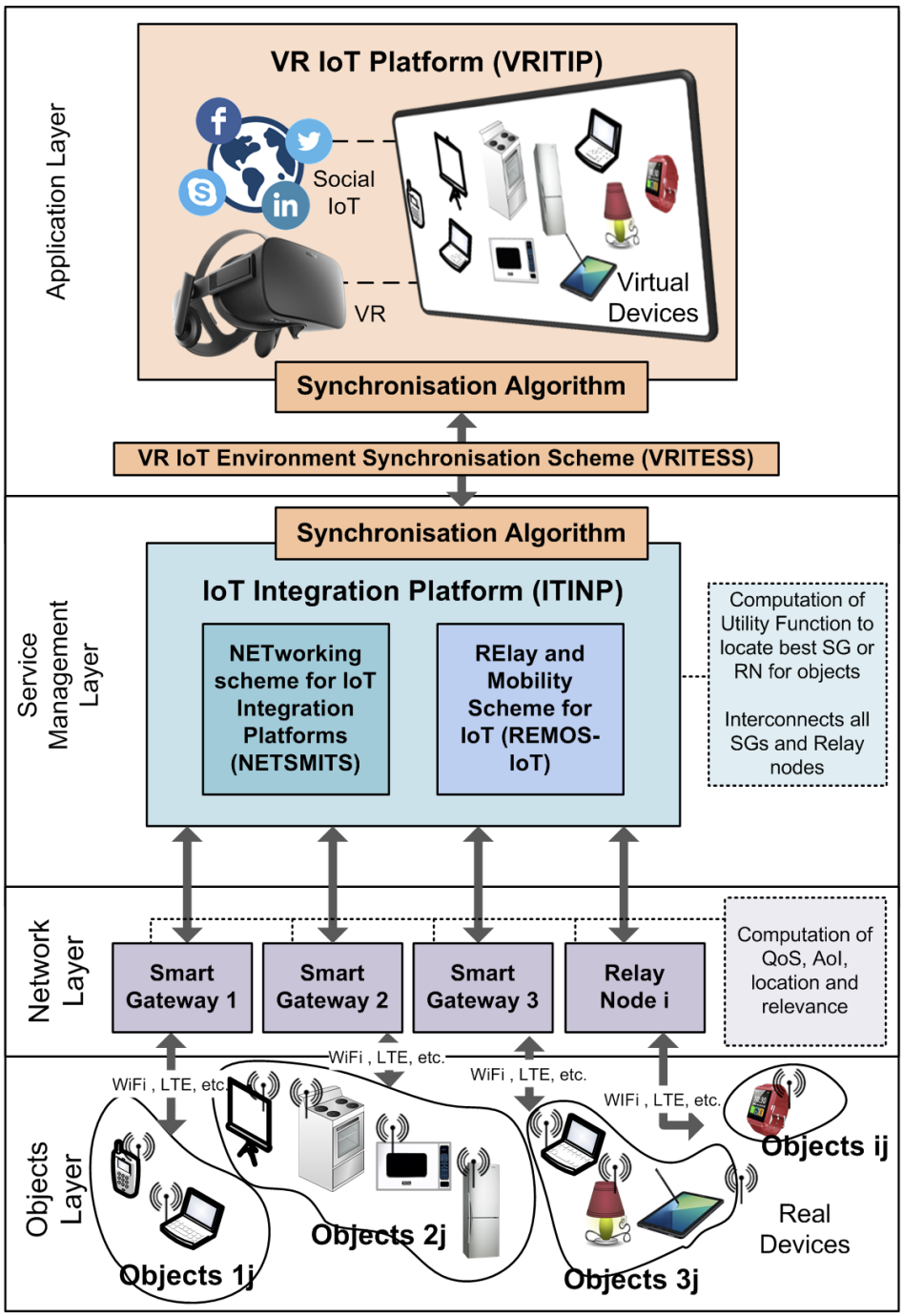


Figure 4.1: Solution Architecture

provide statistics. Smart gateways' data is sent to ITINP, which is located at the Service Management layer. This layer is a bridge to the Applications layer, where software applications can interact with users or with other machines. Finally, the decision-making is done at the Business Layer via Big Data solutions; however, the

contributions of this thesis do not involve this layer. The Service Management layer is also responsible for controlling the network intelligently, integrating different types of data i.e. multimedia data, sensor data. ITINP performs these tasks in order to maintain high network performance levels.

4.1.2 IoT Objects

IoT objects (i.e. wireless devices, sensor based devices, appliances, etc.) provide one or more services to other objects (e.g. user devices) and are initially connected to the closest smart gateway through a short range wireless protocol (e.g. WiFi) in order to exchange data. However, if an IoT object is more relevant to another object cluster in a distant area (i.e. is associated with high number of requests from that area), it will be attached to a smart gateway serving the new area using long range wireless connectivity (e.g. LTE). This approach decreases the load of the original gateway enabling improved performance for the other object communications. Each object attached to a smart gateway (SG_i) is identified via an ID (O_{ij}) and provides one or more services S_{ijk}, where *i* identifies the gateway, *j* indicates the object and *k* specifies the service.

4.1.3 Physical and Logical Clusters

As already mentioned, every IoT object is connected to a smart gateway via short range (e.g. WiFi) or long range (e.g. LTE) wireless communication channel. Initially, objects belong to the closest smart gateway, creating a physical cluster, where the distance helps define the cluster. However, some objects' offered services might be more accessed by objects from other clusters and in this scenario the current smart gateway is only being used as a bridge for communications. In order to prevent such a scenario which affects the performance of communications, each object is reassigned regularly to the most relevant smart gateway. If the associated smart gateway is distant, long range wireless solution such as LTE will be used for object-smart gateway communications, creating a logical cluster. A logical cluster extends the physical cluster concept by including objects located at various locations with which other objects in this cluster interact more often (i.e. provide or consume services).

4.1.4 Smart Gateways

A smart gateway is responsible for receiving and sending data from/to objects, managing admission control, aggregating data and storing performance-related information in form of scores. One of these scores, which is related to **QoS metrics**, is calculated by an algorithm that considers throughput, delay, and loss and is aggregated both at the level of objects and smart gateways. Another algorithm calculates objects' **relevance** scores, considering the percentage of requests coming from different gateways. Objects **distances** are also measured, so moving devices can be supported and reconnected quickly, as the smart gateways are aware of objects' locations. The end-to-end delay (e.g. the time for a message to be sent from a device to the destination device, considering all hops) is another score measured, in the form of the **Age of Information** (AoI) score, which is useful for analysing the overall quality of network delivery.

Objects send and receive data to/from other objects via the smart gateway they are currently attached to. If an object is trying to reach another object that is located in a different smart gateway, it will be accessed through ITINP, which is accessible from all smart gateways. All smart gateways send their own scores to ITINP and regularly the measure QoS, relevance, location and AoI scores of devices for re-clustering of low performing objects.

4.1.5 IoT Integration Platform (ITINP)

ITINP is a cloud-based server platform developed along with the contributions of this thesis. ITINP is responsible for creating and managing optimal clusters of objects and assigning them to the best smart gateway. In order to accomplish this task, an algorithm that considers QoS scores, object relevance, location and AoI (all provided by the gateways) is responsible for associating objects to the most suitable smart gateway, in order to establish logical clusters and improve the gateway communication performance. As ITINP is aware of smart gateway locations, if an object cannot reach the most suitable gateway through a short range wireless connection, the object will be requested to use a long range wireless technology to attach itself to the selected gateway.

4.1.6 VR IoT Platform (VRITIP)

VRITIP is the platform which represents real devices in a virtual environment and was created to deploy the architecture proposed on this thesis. VRITIP controls and presents to users a user-friendly virtual version of IoT devices, and it is also responsible for keeping these virtual devices updated with the latest alterations made in the real world. VRITIP sends alterations made in the virtual world to ITINP, which controls the real world devices. The virtualised translation of IoT services in VRITIP can be accessed through interfaces that have high user acceptance, such as virtual reality devices and social networks.

4.2 Proposed Solutions

The solutions deployed on top of ITINP and VRITIP are the main contributions of this thesis. These solutions were designed to improve network quality, support device mobility, increased number of devices and improve user experiences when operating IoT systems.

4.2.1 NETworking Scheme for sMart IoT gatewayS (NETSMITS)

The NETworking Scheme for sMart IoT gatewayS (NETSMITS) is an innovative IoT solution that supports an increased number of inter-communicating IoT objects and improves quality of their offered services. NETSMITS uses an architecture that includes IoT objects, smart gateways and operates with ITINP. The IoT objects offer and consume diverse services including video streams, the smart gateways inter-connect the IoT objects and ITINP manages the smart gateways. Tests indicate that NETSMITS is able to avoid packet loss, allow the highest throughput per device, greatly reduce delay and substantially increase the number of devices served.

4.2.2 REMOS-IoT (RElay and MObility Scheme for the Internet of Things)

REMOS-IoT (RElay and MObility Scheme for the Internet of Things) adds support for mobile objects by including in its algorithms metrics such location, age of information, QoS, and relevance. The main goal of REMOS-IoT is to provide the best connectivity to objects by assigning them to the optimal IoT smart gateway in the

IoT network or using available devices as relay nodes. The REMOS-IoT architecture contains IoT devices, smart gateways and it works with ITINP. Tests show that REMOS-IoT is able to provide improvements in connectivity to devices, increasing their performance and QoS.

4.2.3 VR-IoT Environment Synchronisation Scheme (VRITESS)

The VR-IoT Environment Synchronisation Scheme (VRITESS) bridges VRITIP and ITINP, connecting devices in both real and virtual worlds. VRITESS allows users to seamlessly operate IoT devices (e.g. beacons and sensor functions in single-board computers such as the Raspberry Pi) in a VR environment, using a simpler interface that supports interaction with the devices. When operating a device in the virtual environment, users' decisions will synchronously be shown in the virtual environment and will also affect real devices, with as minimum latency as possible.

4.2.4 Social IoT Integration

The devices virtualisation possible with VRITIP also provide a framework for another type of interactions with devices. The Social Internet of Things (SIoT) concept which has emerged recently, adds social networking aspects into IoT. Social networks APIs can be connected to cloud platforms and VRITIP, allowing discovery of connected objects and trust between connected objects and users that are 'friends' within the SIoT platform. It is expected that users can operate their devices in a simple and easy way, with minimal latency.

4.3 Chapter Summary

This chapter presented the overall architecture and functionality of the layers that the contributions of this thesis lie on. The roles of IoT devices, smart gateways, ITINP and VRITIP were also defined, aiming to increase IoT networks performance through smart clustering and improve user experience with innovative ways of operating IoT devices with VR and social networks.

The next chapters present details of NETSMITS, REMOS-IoT and VRITESS, solutions that run on top of the architecture containing ITINP and VRITIP, which were presented in this chapter.

Chapter 5

NETSMITS - NETworking Scheme for sMart IoT gatewayS

This chapter introduces NETSMITS, the NETworking Scheme for sMart IoT gatewayS, an innovative IoT solution that improves the performance and quality of IoT services [220], [221]. As described in the previous chapter, NETSMITS uses a layered architecture that consists of IoT objects, which provide diverse services including video applications and CCTV monitoring, smart gateways, which support IoT object network connectivity and a cloud-deployed ITINP employed for resource management. NETSMITS introduces an innovative algorithm that uses QoS and service relevance metrics in order to efficiently cluster the inter-communicating IoT objects, attach them to the most suitable smart gateway and improve the performance of their services. Simulation-based testing shows how NETSMITS outperforms an existing state-of-the-art solution for delivering video services in terms of QoS metrics such as packet loss.

5.1 Problem Statement

The increasing number of devices in IoT networks and the distribution of several gateways using different technologies to access the Internet are a challenge for a large unified IoT network, such as the smart home scenario illustrated in Figure 5.1. The heterogeneity of devices and the distribution of services in the network are additional challenges, because certain devices can affect the performance of the network while other devices may not be connected to the optimal gateway at a certain time. Therefore, a scheme that considers device performance and smart clustering is necessary, resulting in the development of NETSMITS.

NETSMITS goals are to employ and assess the values of QoS and Relevance metrics

at the level of devices and gateways connected to ITINP, in order to perform smart clustering of devices. Through smart clustering, NETSMITS aims to support to an increased number of inter-communicating IoT objects while maintaining higher quality of their offered services.

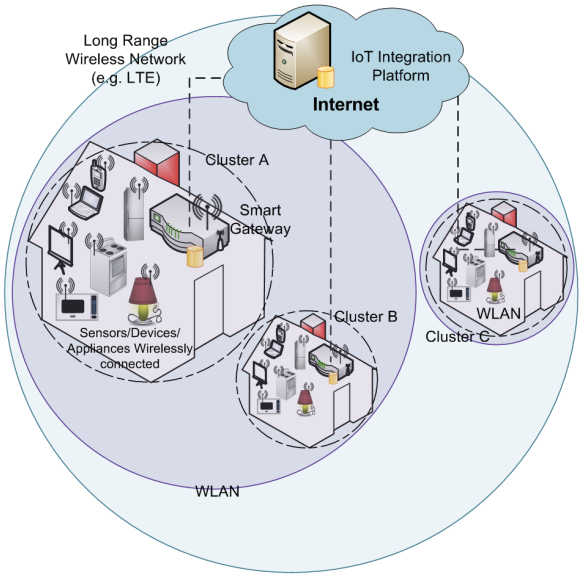


Figure 5.1: Smart Home IoT Network

5.2 NETSMITS Design

NETSMITS contains a clustering scheme for IoT objects based on their services. An algorithm deployed in the cloud-based server – ITINP – is supplied with metrics retrieved from smart gateways to find the most relevant and better performing gateways available for the IoT objects, creating and managing the logical clusters. The algorithm will use WiFi or LTE in order to increase the number of devices connected in clusters and improve the QoS metrics associated with the services provided by these devices. This algorithm also receives a list of the low performing devices based on utility functions described next.

5.2.1 Utility Functions

NETSMITS employs two scores in its algorithm: QoS and Relevance scores, computed based on utility functions. For better understanding of the functions, algorithmic notations are shown in Table 5.1.

Table 5.1: NETSMITS Algorithmic Notations

Symbol	Definition
M_x^{ijk}	Value of QoS metric *x* of a service S_{ijk} (e.g. loss)
R_{ij}	Relevance score of an object *ij*
i	ID of a Smart Gateway
j	ID of an Object
k	ID of a Service
ITINP	IoT Integration Platform
SG_i	Smart Gateway *i*
O_{ij}	Object *ij*
S_{ijk}	Service *ijk*
P_i	Performance score of a smart gateway *i*
P_{ij}	Performance score of an object *ij*
P_{ijk}	Performance score of a service *ijk*
R_{ij}	Relevance score of an object *ij*

The **QoS score** is used for admission and performance control (i.e. a smart gateway with poor performance will request ITINP connectivity to a better performing gateway for new objects) and for ranking of objects (i.e. the objects with low performance are identified). The QoS score associated with an object is calculated by normalising the sum of the normalised values of each metric measured in the service provided by the object – see equation (5.1).

$$P_{ijk} = \sum_x \left(\frac{M_x^{ijk}}{\max_x (M_x^{ijk})} \right) \tag{5.1}$$

Then, an average of the service scores compose the QoS score of an object – see eq. (5.2). The formula in eq. (5.1) is also used to calculate the QoS score of a smart gateway.

$$P_{ij} = \underset{k}{avg}(P_{ijk}) \tag{5.2}$$

The **Relevance score** is computed from an array of tuples composed by the gateways that have communicated with an object and the percentage of packets exchanged between each gateway and the object. For example, a device O_{11} that has exchanged messages with two smart gateways has sent and received a total of 200 packets from devices in the same gateway (SG_1) and has sent and received a total of

400 packets from objects associated with a different gateway (SG_2). Therefore R_{11} = {[SG_1, 33.3%], [SG_2, 66.6%]}.

The research goal in this area is to find a solution that maximises the utilities P_{ij} and R_{ij}. However, an analytical solution cannot be found. NETSMITS employs a good, but not optimal, solution with the use of a heuristic clustering approach, iteratively increasing performance of devices and smart gateways. This can be applied in real time and has low overhead.

5.2.2 Major Architectural Components

Figure 5.2 illustrates NETSMITS' block level components, detailing the architecture described in chapter 4.

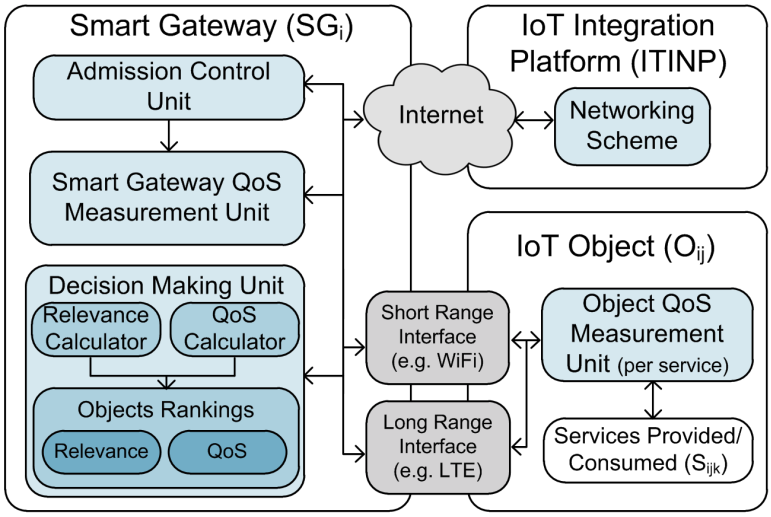


Figure 5.2: NETSMITS Components

The components are the IoT Integration Platform (ITINP), which is located in the Service Management Layer of the architecture; Smart Gateways, which are located in the Network Layer; and the IoT Objects, which are located in the Objects Layer. Each component has units with different tasks, which are described next.

5.2.2.1 IoT Objects' QoS Measurement Unit

The IoT objects' QoS Measurement Unit collects QoS metrics per each service provided by the object (if the object is capable of running the algorithm). IoT devices provide one or more services, for instance, a single-board computer may be used to measure temperatures, perform motion-detection and control lighting, at the same

time, retrieving data and sending it back to gateways. Therefore, QoS metrics must be retrieved per each offered service. Values of these metrics are normalised and compose the QoS score, as presented in equations 5.1 and 5.2. Throughput, delay and packet loss ratio are considered as QoS metrics in this thesis.

5.2.2.2 Smart Gateways' Admission Control Unit

The Admission Control Unit procedure is triggered whenever a new object tries to register to a smart gateway. The Smart Gateways' Admission Control Unit verifies if the gateway can support communication with a new object. The QoS score of the gateways (P_i) are accessed before a new object is attached to the gateway, and if the score demonstrates the gateway is performing poorly, the new object is associated to the next most suitable gateway, identified by ITINP. ITINP maintains scores of gateways, and low scores represent gateways with quality degradation (e.g high delays and packet loss).

5.2.2.3 Smart Gateways' QoS Measurement Unit

The Smart Gateways' QoS Measurement Unit is similar to the IoT objects' QoS Measurement unit, retrieving the QoS metrics (per service) for the objects that are not capable of running the algorithm themselves. Values of these metrics normalised and compose the QoS score, as presented in equations 5.1 and 5.2. This unit also collects the smart gateways' related QoS metrics, used for admission control (i.e. low-performing gateways are avoided for new devices). Throughput, delay and packet loss ratio are considered as QoS metrics in this thesis.

5.2.2.4 Smart Gateways' Decision Making Unit

The Smart Gateways' Decision Making Unit is in charge of computation of QoS and Relevance scores, ranking of objects and deciding if they should be associated with another smart gateway or not. The QoS score calculator uses the utility functions from equation 5.1 and equation 5.2 for processing the QoS metrics related to devices. The Relevance score computation considers the number of packets exchanged with objects associated with diverse gateways. The scores are kept in a gateway database, which also stores the maximum metric values for the normalisation process (i.e $\max(M_x)$), which employs equation 5.2.

5.2.2.5 ITINP Networking Scheme

The ITINP Networking Scheme deploys an algorithm (see Algorithm 5.1), which receives a list of poor performing objects and attaches them to better suitable gateways. The algorithm is triggered by the gateway Decision Making Unit, and the execution frequency can be adjusted according to the type of devices in a gateway. The algorithm is also triggered when new devices try to connect to the gateway and when the gateway is experiencing congestion and performing badly (e.g. devices experiencing loss or high delays).

Algorithm 5.1 Attaching Low QoS-Scoring Objects

Require: List of P_{ij} and R_{ij} per O_{ij} in SG_i
foreach *(O_{ij})* **do**
  if *(SG_i **is not** most relevant SG **and** most relevant SG P_i >= SG_i P_i)* **then**
    most relevant SG.attach(O_{ij})
    SG_i.detach(O_{ij})
  end
  if *(O_{ij} **has** low P_{ij} **and** most relevant SG = SG_i)* **then**
    next most relevant SG.attach(O_{ij})
    SG_i.detach(O_{ij})
  end
end

Algorithm 5.1 attaches low-performing objects (O_{ij}) to other available well-perform-ing gateways, matching devices with the most relevant gateway (i.e. the gateway containing most devices consuming/providing services to the object). The algorithm initially receives from the Decision Making Unit a list with the lowest performing devices contains their performance scores (P_{ij}) and relevance scores (R_{ij}). The algorithm tries to attach devices to their most relevant gateway, when devices are not yet attached to them. The performance of the most relevant gateway (P_i) is also considered, as devices will not be re-attached if the most relevant gateway has poor performance. In relation to devices performing poorly, based on their performance scores, the algorithm tries to attach these devices to better performing gateways, locating the next most relevant gateway if the current one is the most relevant.

The components and algorithm presented in this section aim to improve device clusters in order to allow gateways to perform better and accommodate more devices. The communication process between objects, gateways and ITINP is presented in the next section.

5.3 Components Communications

The sequence diagram in Figure 5.3 presents an example of communications among objects, smart gateways and ITINP running NETSMITS.

The scenario presented in the sequence diagram shows that after objects in gateway 2 request data very often from an object in gateway 1, NETSMITS considers that the object should be re-attached to gateway 2.

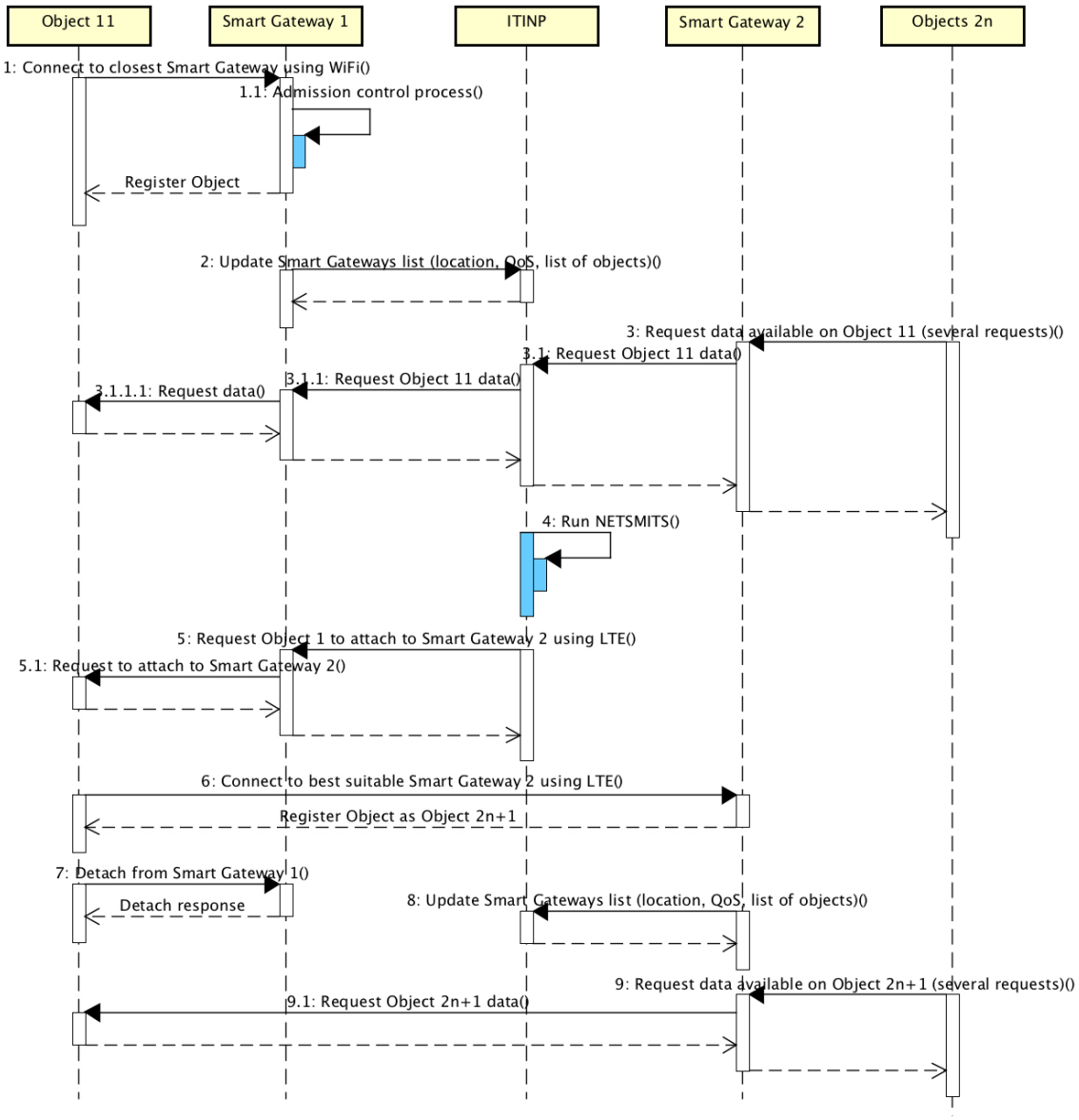


Figure 5.3: Sequence Diagram - Components communications

Object 11 (which means smart gateway 1 object 1) is located near gateway 1 through a WiFi connection. The object is registered, as in the admission control process, the gateway verifies it has the resources for connecting the device. ITINP receives the updated list of devices attached to gateway 1, as well as their locations and QoS metrics.

Several objects from gateway 2 (i.e. objects 2n) start consuming services provided by the object 11. These requests go through the gateway 2, passing through ITINP, and gateway 1, before reaching object 11.

Once the NETSMITS algorithm is executed, object 11 is selected to be attached to gateway 2, as its relevance score in relation to gateway 2 is higher than the score of its relevance in relation to gateway 1. This happens as the majority of requests are coming from devices located in gateway 2.

Object 11 is connected to gateway 2 using an LTE connection, meaning the WiFi connection previously used could not reach gateway 2. Once the new connection is established, the device is disconnected from gateway 1. The object receives a new ID, placing it after the last used ID (i.e. object 2n+1). ITINP, then, receives the updated list of devices attached to gateway 2, as well as their locations and QoS metrics.

The number of hops is decreased, as objects in gateway 2 can now communicate directly with the object now attached to gateway 2, without having to communicate through ITINP and another gateway before reaching the object.

In order to analyse the performance of NETSMITS, its architectural design and its communication process, the performance analysis and simulation results are presented in the next sections.

5.4 NETSMITS Performance Analysis

This section presents the IoT data collected during the execution of NETSMITS, which will be statistically studied. An analysis of the relationship among the QoS metrics is performed, including a comparison with another QoS statistical study. The analysis aims to identify the impact of network QoS metrics when IoT devices with different network requirements coexist.

5.4.1 Simulation Setup

The scenario modelled is a smart house with many devices initially connected to a local gateway using a WiFi 802.11ac access point, as seen in Figure 5.4. Three of the

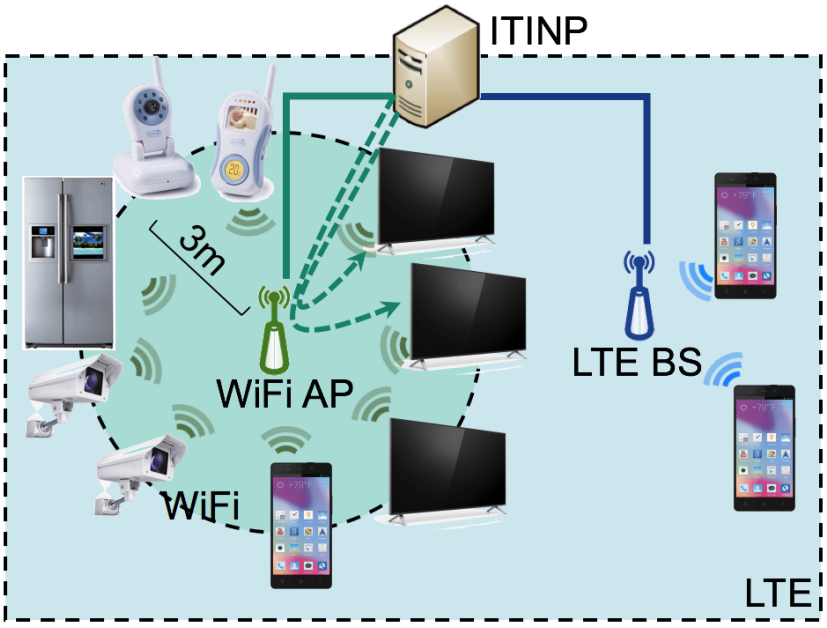


Figure 5.4: Before Running NETSMITS

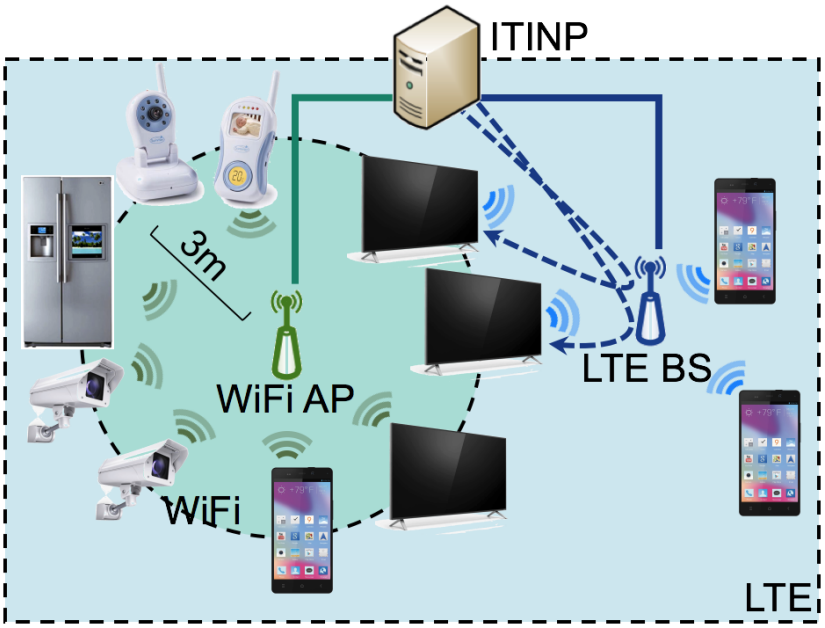


Figure 5.5: After Running NETSMITS

devices are 4K, HD and SD TVs, which receive data at 25 Mbps, 10 Mbps and 4 Mbps, which are optimal values for video streaming in 4K, HD and SD resolutions. The remaining five devices (labelled other) include smart appliances and phones, video monitoring equipment, laptops, etc., which both receive and send data at 1 Mbps, representing services that require lower bitrates. The 1 Mbps bitrate was selected as

Table 5.2: Simulation Setup

Parameter	Value
Simulator	NS-3.24.1
Duration of the Simulation	10s+10s before and after simulation
Distance between nodes and antennas	3m
WiFi Data Rate	40 Mbps
WiFi Standard	802.11ac (40MHz, MCS 9)
LTE eNB Antenna Model Type	Isotropic Antenna Model
LTE Data Rate	100 Mbps
Remote Station Manager	ConstantRateWifiManager
4K Video Bit Rate	25 Mbps
HDTV Bit Rate	10 Mbps
SDTV Bit Rate	4 Mbps
Other devices Bit Rate	1 Mbps

it is a typical rate used in multimedia-enabled IoT systems. This bitrate was used in research works such as the design of a CCTV system for urban rail security [222], an infant monitoring application [223], a human activity monitoring device [224] and a video-conferencing system [225]. NETSMITS regularly checks the delivery performance and suggests moving devices that are impacting performance (i.e. 4K and HD TVs) to other gateways in the realm of ITINP, in order to improve the quality of data delivery for all the devices. Without the presence of another gateway in the same WiFi network, the 4K and HD TVs will be reconnected to a gateway using LTE, as illustrated in Figure 5.5, therefore, devices in this scenario contain both WiFi and LTE interfaces. The goal is to allow the local gateway to accommodate a higher number of devices and maintain delivery quality. The NETSMITS solution has been modelled and evaluated using Network Simulator 3 (NS-3). The model parameters are presented in Table 5.2.

Simulation-based testing can be used for analysing the effects of traffic from high bit rate devices, streaming high quality video which use most of the WiFi gateway network bandwidth, affecting overall system performance.

In order to assess delivery QoS, at millisecond level, performance data is extracted from each device. The metrics measured are packet loss, delay and throughput. Packet loss is the percentage of packets (network communication units) that were sent by a source (e.g. server) and not received by the destination (e.g. IoT device). Delay is the time, expressed in milliseconds, required by the packets to reach their

destinations. Throughput is the successful data rate a device was able to achieve in a certain amount of time, measured in Mbits/s (Mbps).

The data measured in each device is stored in files that contain the time of measurement expressed in milliseconds, bit rate (Mbps), throughput (Mbps), packet loss (percentage), delay (milliseconds), and device type.

The simulation ran for 30 seconds, however data was only exchanged from second 10 until second 20, excluding any potential transitory periods.

5.4.2 Descriptive Statistics and ANOVA Tests

The analysis of simulation data was done in IBM SPSS Statistics, a software that helps to address the entire analytical process, from planning and data collection to analysis, reporting and deployment [226]. SPSS was populated with data generated by the NS-3 simulator.

The tests in SPSS were based on the one-way ANOVA (Analysis of Variance). ANOVA was chosen after authors in [141] proved it is a useful statistical method for analysing QoS. One-way ANOVA was used because the metrics are directly compared against the same metric on different devices.

The 8 simulated devices are classified in four different types in SPSS. Device type 1 is a 4K TV with 25 Mbps video bit rate, device type 2 represents a HDTV with 10 Mbps bit rate traffic, device type 3 is an SDTV with 4 Mbps video bitrate and device type 4 represents general IoT devices with 1 Mbps multimedia bit rate. There were 5 of those general devices in the testbed, and an average of their performance rates was taken. As mentioned in section 5.4.1, devices are initially connected to a WiFi network, and low performing objects are re-attached to an available LTE network in order to improve quality.

Figures 5.6 and 5.7 show the means of packet loss and throughput per device type with error bars indicating a confidence interval level of 95%.

As the bit rate is different depending on the device type, it would not be fair comparing the throughput of the devices, so for the analysis, we normalised the metric by retrieving the percentage of the maximum achievable throughput that the devices were able to achieve.

Table 5.3 shows the ANOVA results, which used the Tukey post hoc test with 0.05 significance. In terms of delay, having a 0.99 significance shows that there is no statistical difference among the results associated with the different device types; however,

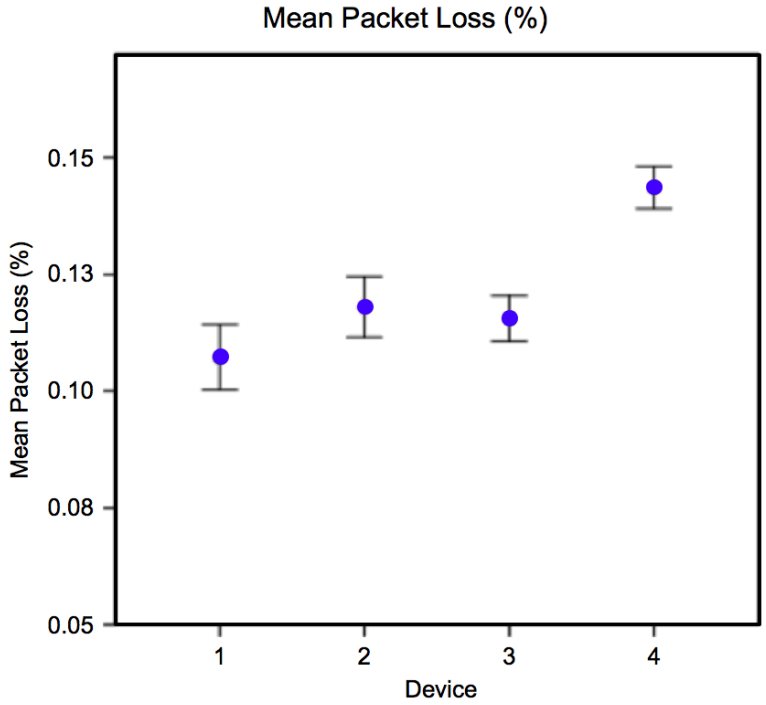


Figure 5.6: Mean Packet Loss. Error bars: Confidence Interval level of 95%

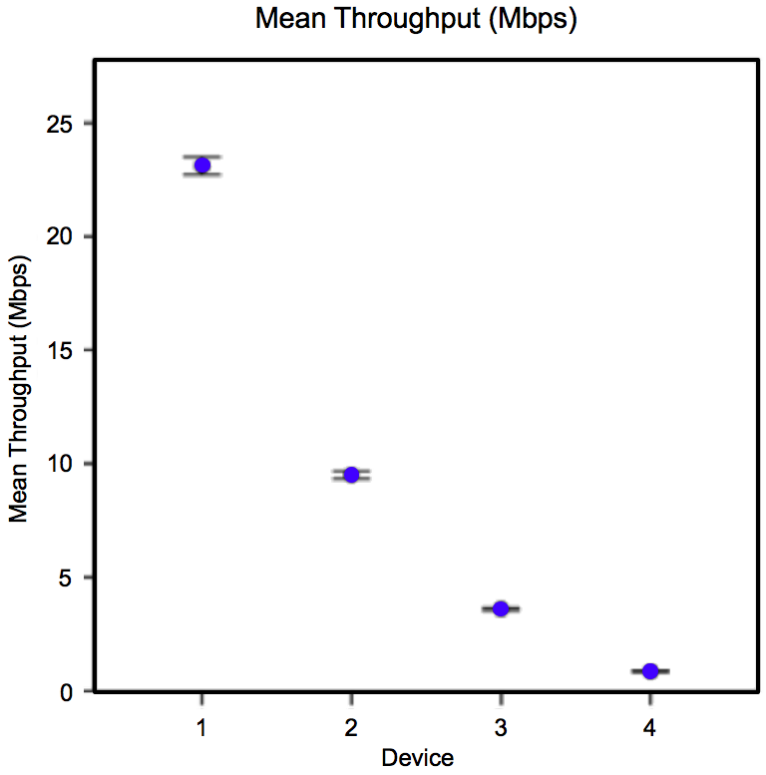


Figure 5.7: Mean Throughput. Error bars: Confidence Interval level of 95%

Table 5.3: ANOVA Tests

		Sum of Squares	df	Mean Square	F	Sig.
Pct. Achieved of Max. Thru. [%]	Between Groups	0.48	3	0.16	16.27	0
	Within Groups	3.977	404	0.01		
	Total	4.457	407			
Delay [ms]	Between Groups	6.02	3	2.007	0.005	0.999
	Within Groups	155194.647	404	384.145		
	Total	155200.667	407			
Packet Loss [%]	Between Groups	0.075	3	0.025	28.76	0
	Within Groups	0.352	404	0.001		
	Total	0.427	407			

for throughput and packet loss, which are below 0.05, there is such a statistical difference. The Tukey method is used to compare all possible pairs of means and is based on a similar distribution of t from the t-test [227].

The multiple comparisons results, presented in Table 5.4, show that some objects had similar behaviour in the network, which might have affected the others.

Device types 2 and 3 had similar results in terms of packet loss, which show that, although all 4 device types had a similar average loss between 10% and 14%, the distribution of loss during the simulation differed. Device types 2 and 3 had no significant difference between packet loss results, and neither device types 1 and 3. Device types 1 and 2 results show a small statistical difference. Device type 4, which has low bit rate, is the one that suffered the most from packet loss and therefore there is a statistical difference between its results and those of the other three devices.

Regarding throughput, the device types that were most affected were devices 3 and 4, again, the ones with lower bit rates. While between device types 1 and 2, and device types 1 and 3 there are no statistical significant result differences, device types 2, 3 and 4 have statistical differences among them. This statistical analysis shows that the high bit rate devices receive higher bandwidth share and therefore benefit in terms of quality.

Comparing this study with the analysis done in [141], similarities can be found. The authors used ANOVA to evaluate the quality of their novel model based on the Response Surface Methodology (RSM). The model uses a set of experiments to obtain optimal deployment parameters in respect to QoS metrics. Similar to the analysis done on NETSMITS data, a large number of nodes affected the overall perform-

Table 5.4: ANOVA Multiple Comparisons

Depend. Var.	(I) Dev.	(J) Dev.	Mean Diff. (I-J)	Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Pct. Achiev. of Max. Thru. [%]	1	2	-0.026	0.013	0.213	-0.062	0.008
		3	0.024	0.013	0.31	-0.011	0.059
		4	0.066	0.013	0	0.03	0.102
	2	1	0.026	0.013	0.213	-0.008	0.062
		3	0.05	0.013	0.002	0.015	0.086
		4	0.093	0.013	0	0.057	0.129
	3	1	-0.024	0.013	0.31	-0.059	0.011
		2	-0.05	0.013	0.002	-0.086	-0.015
		4	0.042	0.013	0.013	0.006	0.078
	4	1	-0.066	0.013	0	-0.102	-0.03
		2	-0.093	0.013	0	-0.129	-0.057
		3	-0.042	0.013	0.013	-0.078	-0.006
Delay[ms]	1	2	0.049	2.744	1	-7.03	7.13
		3	0.069	2.744	1	-7.01	7.15
		4	-0.235	2.744	1	-7.32	6.84
	2	1	-0.049	2.744	1	-7.13	7.03
		3	0.02	2.744	1	-7.06	7.1
		4	-0.284	2.744	1	-7.36	6.8
	3	1	-0.069	2.744	1	-7.15	7.01
		2	-0.02	2.744	1	-7.1	7.06
		4	-0.304	2.744	1	-7.38	6.78
	4	1	0.235	2.744	1	-6.84	7.32
		2	0.284	2.744	1	-6.8	7.36
		3	0.304	2.744	1	-6.78	7.38
Packet Loss [%]	1	2	-0.01	0.004	0.047	-0.021	0
		3	0	0.004	0.187	-0.018	0.002
		4	-0.036	0.004	0	-0.046	-0.025
	2	1	0.01	0.004	0.047	0	0.021
		3	0.002	0.004	0.934	-0.008	0.013
		4	-0.025	0.004	0	-0.036	-0.014
	3	1	0.008	0.004	0.187	-0.002	0.018
		2	-0.002	0.004	0.934	-0.013	0.008
		4	-0.028	0.004	0	-0.038	-0.017
	4	1	0.036	0.004	0	0.025	0.046
		2	0.025	0.004	0	0.014	0.036
		3	0.028	0.004	0	0.017	0.038

ance. On NETSMITS, the large amounts of data consumed affected the performance of devices. In the RSM-based model, the probability of packet loss in a 5 km range varies from 0% for 100 low-power nodes to 60% for 900 low-power nodes.

The findings presented in the analysis presented in this section show that delay is very similar in devices with different bit rate requirements, while throughput and packet loss affect mostly the low bit rate devices, giving an advantage to the high bit rate devices, which is reflected in terms of quality.

The next section will present other scenarios where NETSMITS can add additional devices into the network after re-clustering.

5.5 NETSMITS Study for Increased Number of Devices

The tests presented in this section aim to demonstrate how NETSMITS is able to increase number of devices, while improving their quality, by re-arranging low-performing devices.

5.5.1 Scenarios Description

Four phases are considered, initially with 8 devices, then with an increased amount of devices.

Phase 1 consists of 8 devices with different bitrate requirements, as described in section 5.4.1. These devices are connected to a WiFi 802.11ac access point, and the two devices with highest bitrate (i.e. 25 Mbps and 10 Mbps) are affecting performance of devices, due to their high requirements.

Phase 2 maintains the same 8 devices, however, after NETSMITS re-allocation, the devices affecting performance are attached to an available LTE gateway.

Phase 3 adds 14 extra devices with bitrates of 1 Mbps to the WiFi network, in order to demonstrate the improvements in quality and check if the network can accommodate these extra devices without major impact on performance.

Phase 4 adds 15 devices (1 Mbps each) into the WiFi network, instead of 14 from phase 3, which seems to start affecting network performance.

The details of the above phases are presented next.

Table 5.5: Performance Analysis of NETSMITS

Devices			Phase 1				Phase 2				Phase 3 (SDTV + 19 devices on WiFi)				Phase 4 (SDTV + 20 devices on WiFi)			
	UL/ DL	Bit Rate [Mbps]	Netw.	Packet Loss [%]	Avg. Thru. [Mbps]	Avg. Delay [ms]	Netw.	Packet Loss [%]	Avg. Thru. [Mbps]	Avg. Delay [ms]	Netw.	Packet Loss [%]	Avg. Thru. [Mbps]	Avg. Delay [ms]	Netw.	Packet Loss [%]	Avg. Thru. [Mbps]	Avg. Delay [ms]
4K TV	DL	25	WiFi	8%	23.11	130.3	LTE	0%	25.58	10.00	LTE	0%	25.58	10.00	LTE	0%	25.58	10.00
HDTV	DL	10	WiFi	9%	9.51	130.25	LTE	0%	10.69	10.00	LTE	0%	10.69	10.00	LTE	0%	10.69	10.00
SDTV	DL	4	WiFi	10%	3.60	130.24	WiFi	0%	4.05	20.04	WiFi	0%	4.03	61.37	WiFi	2%	3.9	182.55
Device 1	DL	1	WiFi	0%	0.99	130	WiFi	0%	1.00	20.00	-	-	-	-	-	-	-	-
	UL	1		0%	1.01	0.59		0%	1.01	0.56	-	-	-	-	-	-	-	-
Device 2	DL	1	WiFi	1%	0.97	130.2	WiFi	0%	1.00	20.00	-	-	-	-	-	-	-	-
	UL	1		0%	1.01	0.56		0%	1.01	0.46	-	-	-	-	-	-	-	-
Device 3	DL	1	WiFi	13%	0.86	130.54	WiFi	0%	1.00	20.00	-	-	-	-	-	-	-	-
	UL	1		0%	1.01	0.76		0%	1.01	0.54	-	-	-	-	-	-	-	-
Device 4	DL	1	WiFi	41%	0.59	130.77	WiFi	0%	1.00	20.15	-	-	-	-	-	-	-	-
	UL	1		0%	1.01	0.66		0%	1.01	0.36	-	-	-	-	-	-	-	-
Device 5	DL	1	WiFi	71%	0.29	131	WiFi	0%	1.00	20.97	-	-	-	-	-	-	-	-
	UL	1		0%	1.01	0.59		0%	1.01	0.43	-	-	-	-	-	-	-	-
Avg.	DL	1	-	-	-	-	-	-	-	-	WiFi	0%	1.00	64.43	WiFi	11%	0.88	183.95
Rest of Devices	UL	1	-	-	-	-	-	-	-	-		0%	1.00	2.80		0%	0.99	3.01

5.5.2 Performance Evaluation and Results

The results, available in Table 5.5, show that after employing NETSMITS, the values of the QoS metrics improve, especially on the downlink (DL) of the devices. In phase 1, when all the devices are connected to the WiFi gateway, the packet loss ratio reaches up to 71% in one of the devices, and delay on the downlink is around 130ms.

The highest throughput is not being achieved by any device, either. In phase 2, the innovative NETSMITS connectivity re-allocation happens. NETSMITS selected the 4K TV and the HDTV to be reattached to the LTE gateway. The results of phase 2 show that in the WiFi gateway packet loss is avoided, the highest throughput per device is achieved (Figure 5.8) and the delay is reduced by 85% (Figure 5.9).

Phases 3 and 4 add more devices to the WiFi gateway. Two different tests are considered. Phase 3 demonstrates that 14 extra other devices with 1 Mbps bit rate each (an increase of 63% in number of devices) can be added without major QoS degradation. Figure 5.11 demonstrates that throughputs are not affected in both phases 3 and 4, as devices maintain high throughput.

Phase 4 involves adding 15 extra devices (1 Mbps each) and results in some performance degradations, with an 185% increase of delay, as seen in Figure 5.12.

While phase 3 does not introduce packet loss after the addition of 14 devices, phase 4 generates packet loss in the devices with lower bitrates, after the inclusion of the 15 extra devices. As seen in Figure 5.13, there is an average packet loss of 2% for the SDTV and 11% for the devices with 1 Mbps bitrate in phase 4.

NETSMITS can be compared to a baseline approach in terms of packet loss. The selected baseline solution is the multi-channel Cluster-Based Framework [160] for

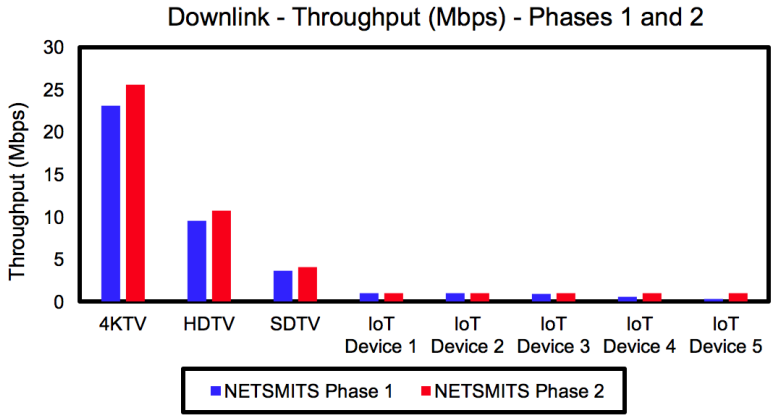


Figure 5.8: Throughput (Mbps). Comparison between NETSMITS phases 1 and 2

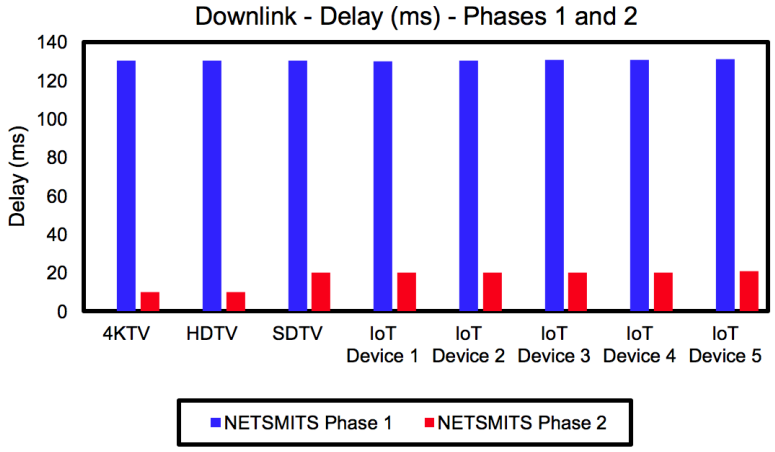


Figure 5.9: Delay (ms). Comparison between NETSMITS phases 1 and 2

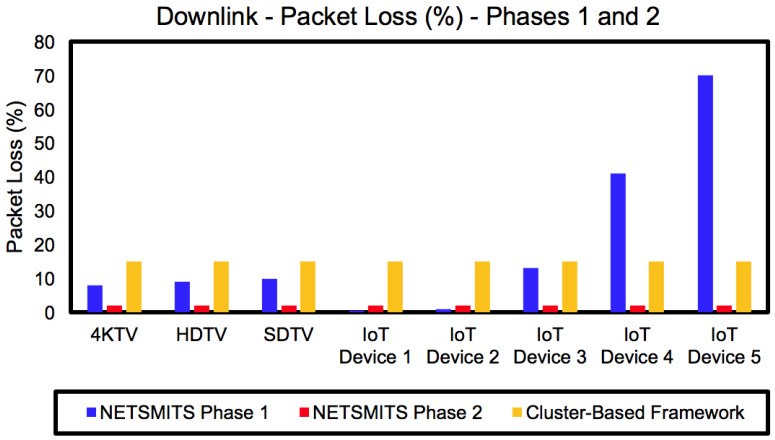


Figure 5.10: Packet Loss Ratio (%). Comparison between the Cluster-Based Framework and NETSMITS phases 1 and 2

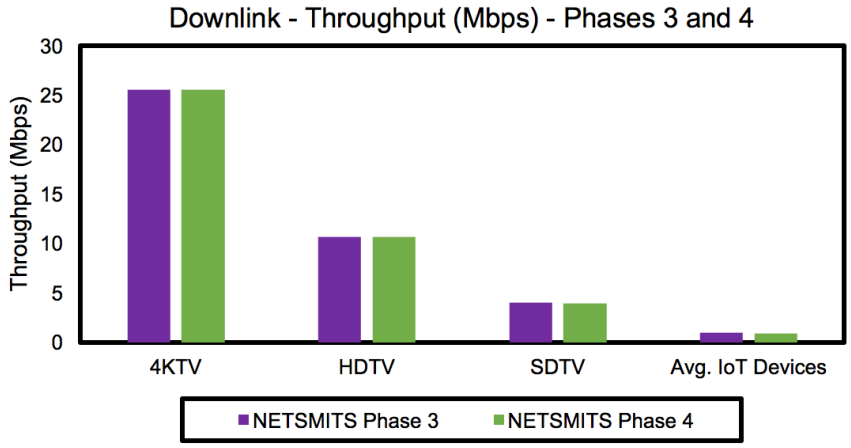


Figure 5.11: Throughput (Mbps). Comparison between NETSMITS phases 3 and 4

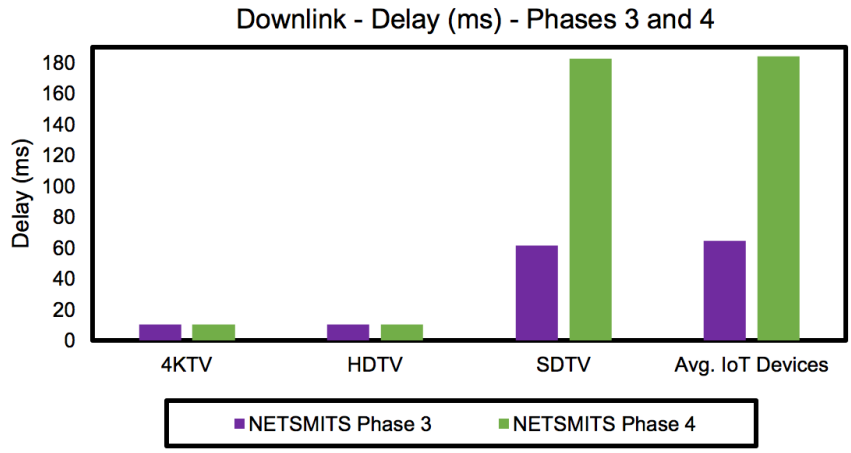


Figure 5.12: Delay (ms). Comparison between NETSMITS phases 3 and 4

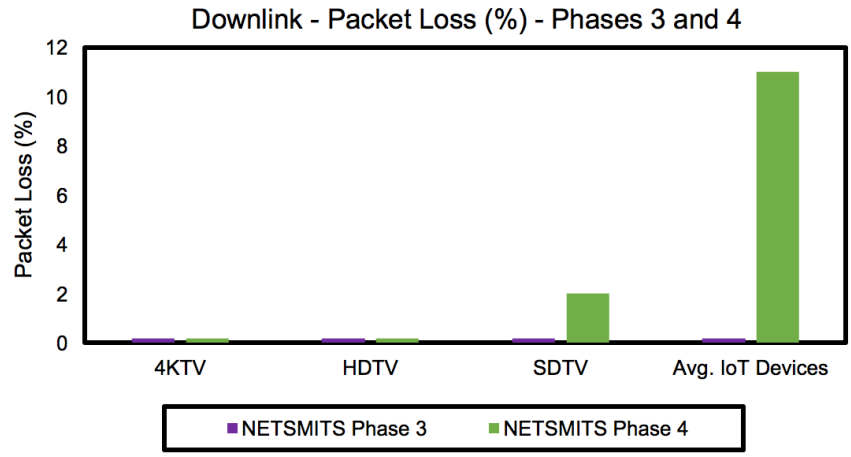


Figure 5.13: Packet Loss Ratio (%). Comparison between NETSMITS phases 3 and 4

mobile Wireless Mesh Networks. The solution considers mesh points that degrades reliability of network. The novel cluster-based framework allows the network to be managed more efficiently with a cluster head, cluster members and cluster gateways. Every cluster uses a different channel for intra-cluster communication, avoiding collisions. Devices are clustered in order to improve QoS metrics in both static and mobile scenarios.

It can be seen how NETSMITS outperforms the Cluster-Based Framework, having packet loss decreased from 15% in the framework, to zero in NETSMITS phase 2 (Figure 5.10). The comparisons were made against the average values of the Cluster-based Framework static approach.

5.6 Chapter Summary

This chapter has introduced the Networking Scheme for Smart IoT Gateways (NETSMITS), which supports increased number of inter-communicating IoT objects and maintains higher quality of their offered services. A statistical analysis of NETSMITS was also presented, demonstrating the effects of different types of devices with different bitrate requirements in the network.

The novel aspects of NETSMITS include architectural components and an algorithm that extracts QoS and relevance performance metrics from devices and their services, and based on devices and gateways performance, the algorithm re-clusters the devices in appropriate available gateways, in order to increase overall network delivery quality.

The benefits of NETSMITS are demonstrated by the results obtained, allowing devices to achieve very high throughput with minimum delay and loss, and allowing the network to accommodate extra devices.

The next chapter presents REMOS-IoT, which works along with NETSMITS with more performance metrics, allowing mobility and relays to be considered in the re-clustering process.

Chapter 6

REMOS-IoT - RElay and MObility Scheme for the Internet of Things

This chapter introduces the RElay and MObility Scheme for the Internet of Things (REMOS-IoT) [228], [229], which enhances IoT architectures with algorithms that constantly monitor under-performing devices, and improve mobile IoT device connectivity performance. Devices such as wearables, sensors, drones, smart vehicles, etc., need constant connectivity despite their moving patterns and therefore, an IoT architecture should consider both Quality of Service (QoS) and mobility. D2D allows devices to communicate directly to share content and functionality, such as access to the Internet. REMOS-IoT uses a D2D communication-based architecture and algorithms, which support an improvement of device metrics such as packet loss, delay and throughput, in data exchange services between mobile IoT devices, working along with NETSMITS presented in the previous chapter for re-clustering devices. Simulation-based testing showed how performance of devices increases in different scenarios, demonstrating the efficiency of the proposed architecture and algorithms.

6.1 Problem Statement

The heterogeneity of IoT networks introduces several challenges in relation to real-time coordination, data storage, network performance, concurrency, mobility patterns, QoS etc. [6]. Innovative solutions can improve the values of some metrics during data delivery, especially in terms of QoS metrics, such as delay, packet loss and throughput. In order to assess and enhance QoS, solutions employ several approaches, including adaptation schemes [230], [231], scheduling techniques [128], [232], clustering algorithms and other innovative communication approaches [233], [234].

Device-to-device (D2D) communications, which allow direct mobile device intercommunication, benefit IoT devices when they cannot reach any base station or there is limited bandwidth available. Other benefits of D2D include energy conservation, QoS improvements, and load balancing optimisation, due to certain devices relaying data via other devices [235], [236]. D2D has also been used in cooperative vehicular networks [237]. Architectures that enable IoT to benefit from D2D communications have been proposed in [164]–[166].

Many types of IoT devices are constantly moving, and there is a need for solutions that provide connectivity when these devices cannot reach any base station/access point or there is limited bandwidth available. Metrics such as RSSI can be used for detecting if devices are moving away from a gateway, so that the solution can proactively offer connectivity alternatives for mobile devices. Devices performance is also a challenge in IoT networks, which can be addressed through a smart arranging of devices in the network, making good use of available resources, such as nearby gateways.

The RElay and MObility Scheme for the Internet of Things (REMOS-IoT) works along with NETworking Scheme for sMart IoT gatewayS (NETSMITS), presented in chapter 5, by providing mobility and relay alternatives to devices, as NETSMITS supports static smart devices. REMOS-IoT's main goals are to analyse smart devices metrics to *provide the best connectivity to these objects by assigning them to the optimal IoT smart gateway in the IoT network, and support an increase of QoS for data exchange services between mobile IoT devices.*

6.2 REMOS-IoT Design

The REMOS-IoT architecture and its major components are illustrated in Figure 6.1, which also shows an example of an usage scenario in the context of a mobile urban environment. REMOS-IoT contains IoT devices (or objects), smart gateways and a cloud-based IoT Integration Platform (ITINP). IoT objects provide and consume services from other objects in the REMOS-IoT architecture. IoT objects are interconnected via the smart gateways. Smart gateways form device clusters in order to improve device intercommunication, identify low-performing and moving devices and manage admission control. ITINP manages smart gateways and relay nodes, and increases clusters performance by computing objects performance metrics then rearranging low-performing and moving objects into suitable smart gateways or relay

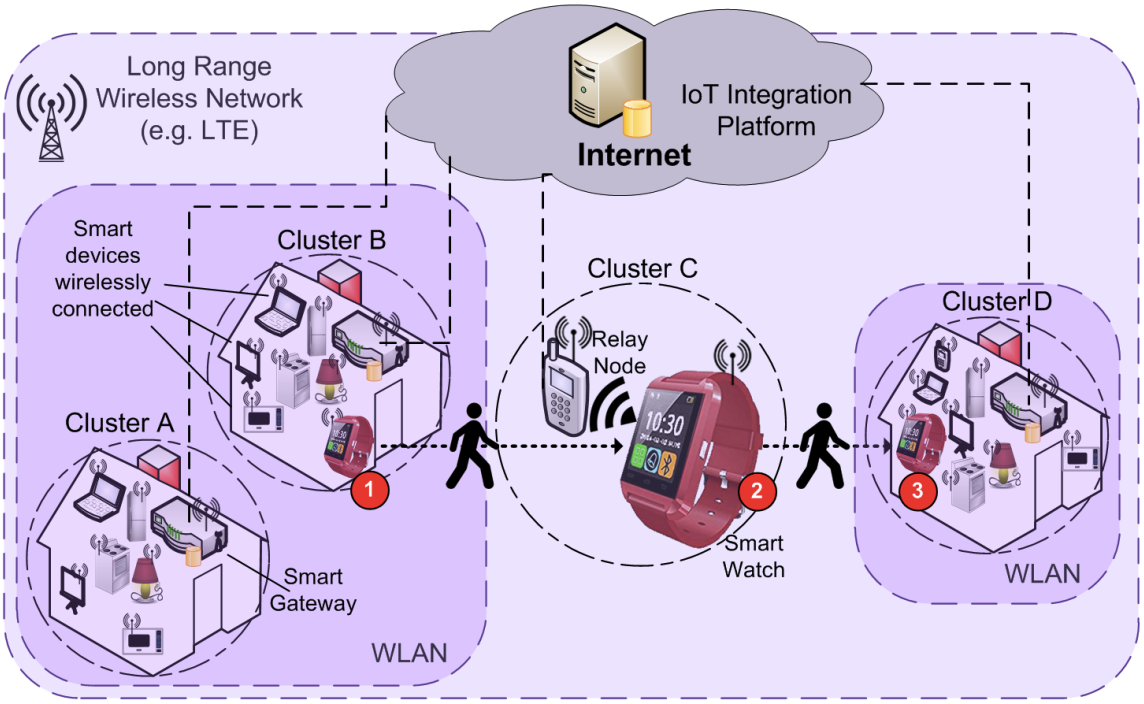


Figure 6.1: REMOS-IoT Architecture

nodes.

The REMOS-IoT algorithms aim to improve the performance of communication by connecting devices to the best available smart gateways or to selected relay nodes. These algorithms consider diverse factors including distance between devices and smart gateways, and estimation of communication performance based on QoS metrics such as packet loss, delay and throughput. They also take into consideration device mobility, communication relevance and age of information. It is expected that by the use of REMOS-IoT, devices present a decrease in packet loss and delay and achieve higher throughput, while maintaining connectivity of moving devices.

Although other IoT architectures have also considered QoS, to the best of authors' knowledge, they have not employed parameters such as object mobility, relevance and age of information. The proposed REMOS-IoT architecture and solution integrate these and QoS metrics and use them in innovative algorithms in order to improve the overall IoT network performance.

The scenario presented in Figure 6.1 illustrate a situation where the REMOS-IoT architecture can be employed. A mobile user with a smart watch is moving from one wireless local area network (WLAN) (step 1 in Figure 6.1) towards another WLAN (step 3). However, while moving, the device needs to use a relay node to remain

connected (step 2 in Figure 6.1), as it is out of WLAN cover.

6.3 REMOS-IoT Architecture

The REMOS-IoT architecture consists of clustered IoT devices, smart gateways, and the IoT Integration Platform (ITINP), a cloud-based server application. This architecture extends NETSMITS presented in chapter 5 by providing mobility support, D2D communications and enhanced QoS awareness via the following metrics: age of information, location, device relevance and performance (i.e. a score combining loss, delay and throughput).

REMOS-IoT architectural components (i.e. IoT devices, smart gateways and ITINP) contribute to different layers in the five-layer architecture proposed in [121], [219].

Figure 6.2 illustrates the major REMOS-IoT components in the context of these five IoT architectural layers. The *Objects layer* contains the IoT objects, such as appliances, smart sensors, etc, and supports basic D2D communication links between them. The *Network layer*, which contains the smart gateways, enables further communication support between objects, complementing D2D communications. The smart gateways cluster local IoT objects around them, and try to deploy solutions to improve overall communication performance by improving network conditions. The *Service Management layer* includes ITINP, which receives performance data from the smart gateways. As a proper Service Management layer component, ITINP performs smart control of network communications and involves management of the exchange of multiple data types such as multimedia content, sensor feedback, etc. The *Applications layer* deploys user applications and allows for remote communications in the context of various user services. The *Business layer* contains Big Data applications, which are mainly used for policy-related decision-making. REMOS-IoT does not have components at upper two IoT architectural layers.

The following sub-sections present in details REMOS-IoT and its components.

6.3.1 REMOS-IoT Main Architectural Components

IoT devices such as wearables (e.g. smart watches, wrist bands), smartphones, sensors, robots, drones, smart appliances, etc., offer a wide range of services for users and other devices. In a cluster of objects, certain objects can be more relevant to others, depending on the amount of service requests received within the cluster. Objects

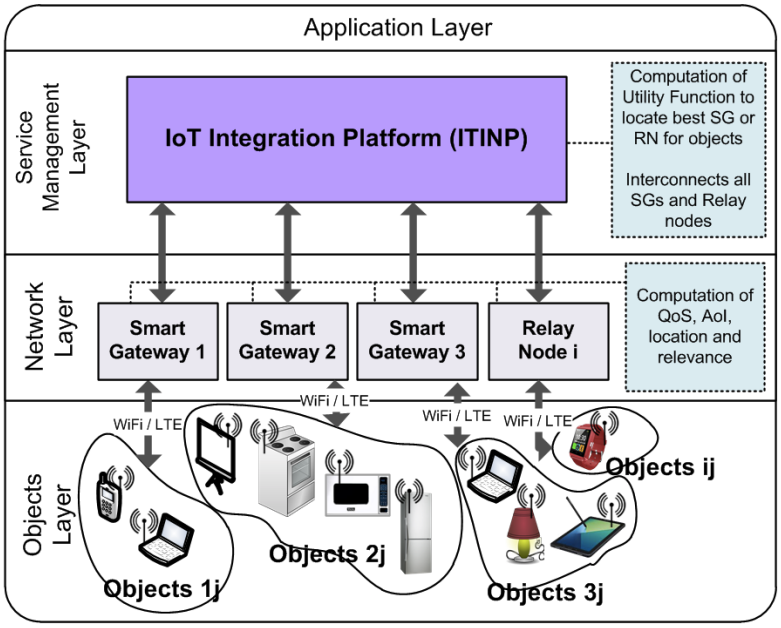


Figure 6.2: REMOS-IoT Layered Architecture

can avail themselves of long or short range connections (e.g. Long-Term Evolution (LTE) or WiFi) according to positioning, connection availability, performance and relevance of the object to a cluster, which can be physically near of far from the object. Objects connected to a smart gateway or relay node (SG/RN_i) receive an ID (O_{ij}) and are able to provide one or more services S_{ijk}, where *i* indicates the gateway or relay node, *j* identifies the object and *k*, the service. Objects are initially attached to the closest smart gateway, creating a cluster defined by distance. Object services requested often by devices from other clusters, might indicate that the object is more related to another cluster. In this scenario, the object might be using the current gateway solely as a bridge for communications, affecting performance. Therefore, it is important a regular reassignment of each object to the most relevant smart gateway. When the most relevant smart gateway is located outside of that range of the object, a long range wireless network (e.g. LTE) will be selected for smart gateway-object communications, creating a logical cluster. REMOS-IoT contains an algorithm for mobility support, balancing location and performance, while considering the availability of relay nodes (e.g. smartphones and other objects able to share internet access) for networking mobile IoT devices.

Smart gateways interconnect IoT objects, manage admission control, calculate and store performance-related information and are responsible for decision-making in relation to objects. IoT objects communicate to each other through the smart gateways

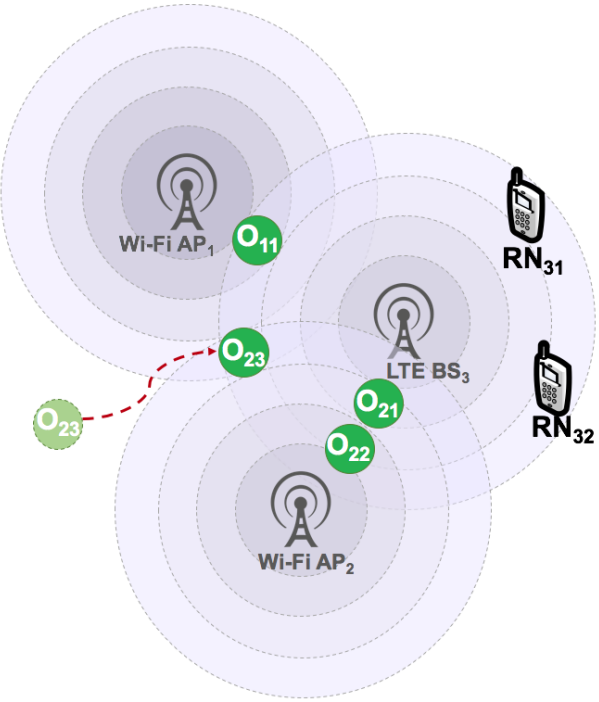


Figure 6.3: Smart Gateway Admission Control Scheme

that interconnect them. In order to allow devices to access services provided by objects connected to other smart gateways, the cloud-based ITINP is used, as it interconnects all smart gateways from REMOS-IoT network. Smart gateways are aware of devices' locations, supporting their mobility. Locations are used in the clustering algorithms and for admission control. As seen in Figure 6.3, the object O_{23} is moving in the direction of an area covered by two access points and a base station. In order to determine which access point or base station will reply to the connection request of the object, gateways calculate the score that elects the best gateway to reply first. For example, object O_{21} is closer to BS_3, but according to the object's reputation and BS_3 available resources, the device is better connected to AP_2.

ITINP is a cloud-based server in charge of forming and managing improved clusters of IoT devices, connecting them to the best available smart gateway. ITINP achieves this by computing objects' performance, and by using an algorithm for re-clustering. ITINP has smart gateways' locations, therefore, if a device cannot reach the most suitable gateway using a short range wireless connection (e.g. WiFi), the device will either be requested to connect itself to the new gateway using a long range wireless technology (e.g. LTE), or be requested to attach itself to an available relay node, in order to remain connected to ITINP and the rest of the network.

6.3.2 Components' Units and Algorithms

Each component of REMOS-IoT architecture, i.e. IoT objects, smart gateways and IT-INP, contains units responsible for gathering of QoS information, data processing, and decision making. These units, also used for NETSMITS, were introduced in chapter 5, described in section 5.2.2, and have been extended for REMOS-IoT as illustrated in Figure 6.4 and described in this section.

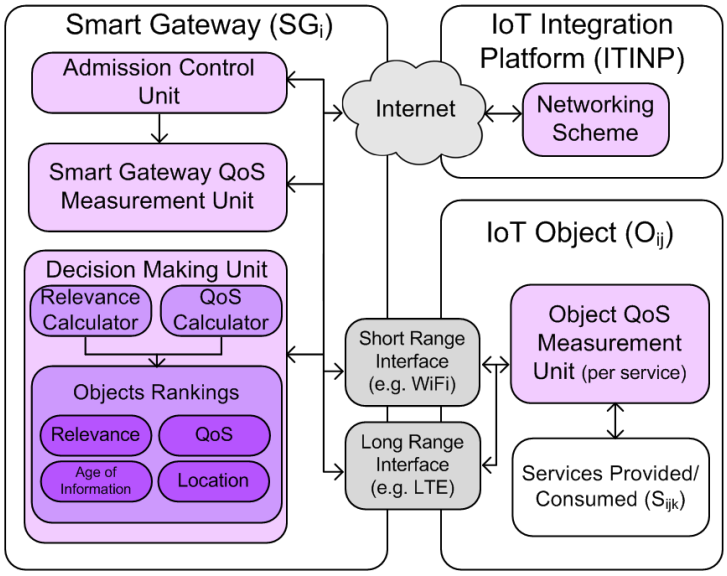


Figure 6.4: REMOS-IoT Units

Each IoT object's QoS Measurement Unit is responsible for collecting regularly QoS data from the services available in the device (in devices able to run the collection algorithm). This data is then sent to the smart gateway, which weights, normalises and averages the values, calculating QoS device scores. The metrics considered for analysis are packet loss ratio, delay and throughput.

The Smart Gateway's Admission Control Unit assesses objects' reputations in order to accept or reject devices. It also checks the gateway performance (computed by Smart Gateways' QoS Measurement Unit) in order to determine whether it can support a new incoming device or not. If it cannot support extra devices, ITINP will identify the next most suitable smart gateway for the device. Locations of devices are also measured in this unit, and then informed to the Decision Making Unit.

The Smart Gateway's QoS Measurement Unit retrieves the QoS metrics of devices that do not have the resources to run the QoS data collection algorithm themselves. These metrics are collected per service. This unit is also responsible for collecting

the QoS metrics from the smart gateways, so the gateway performance can be analysed for admission control of incoming devices. This unit also computes the age of information in relation to the communications between the gateway and devices.

The Smart Gateway's Decision Making Unit is responsible for computing metrics and ranking devices in relation to relevance, QoS, location and Age of Information scores. It also identifies low-performing or moving objects that might be re-assigned to other gateways or relay nodes. Functions presented in eq. (1) and eq. (2), in section 6.3.3, are evaluated in the QoS Calculator unit, whereas the Relevance Calculator computes the number of packets exchanged between devices and gateways, determining which clusters are closely related to a device. The computed scores are stored at the level of the smart gateway, which also stores their maximum values required for normalisation.

The ITINP Networking Scheme deploys two innovative heuristic algorithms, one for object mobility support, and one for re-clustering low-performing objects.

6.3.2.1 Object Mobility Support algorithm

The first algorithm, the *Object Mobility Support algorithm*, is a handover algorithm which extends the scheme introduced in [238] and is described in Algorithm 6.1. The algorithm performs mobility detection, allowing gateways and relay nodes to proactively disconnect and reconnect devices. The mobility support algorithm is triggered when the average RSSI or RSRP (L_{ij}) of devices reach a minimum threshold (set by default to -90 dBm) in relation to their current gateway. The minimum threshold can also be adjusted for networks with different requirements.

The gateway broadcasts a message with the object ID and its location score to neighbouring gateways/relay nodes, and the object that is moving also receives this message. The goal is to receive messages from other gateways and relay nodes to select a good option for the device moving away.

The object with low RSSI/RSRP, then, also sends a broadcast message, so nearby gateways and relay nodes can calculate their distances in relation to the object and verify the reputation score of the device in relation to them. These gateways, which also received a message from the device's current gateway, send a message containing the calculated location and reputation scores in relation to the object back to the current gateway, so the gateway can select the future gateway with the best combination of location and reputation.

The current gateway notifies the target gateway, which attaches the object before

the current gateway can detach it.

As novel aspects, the proposed Object Mobility Support algorithm uses new performance metrics based on device reputation and location as part of an utility function (U_{ij}) in the process of improved gateway selection. The utility function (U_{ij}) is detailed in section 6.3.3. It also makes use of D2D communications with available relay nodes in order to maintain connectivity of mobile IoT devices outside smart gateway range. This enables the possibility of finding either available well-performing gateways or relay nodes to keep device connection alive, when the current communication signal reaches a minimum threshold level.

Algorithm 6.1 Providing Mobility Support for Objects

if *(L_{ij} <= threshold)* **then**
  SG_i **broadcasts** *message* **with** O_{ij} **and** L_{ij}
  SG_i **waits** other *SGs* **and** *RNs* (within O_{ij} range) reply
  O_{ij} **receives** *message* **broadcasted** by SG_i
  O_{ij} **broadcasts** *message* to *SGs* and *RNs*
  SGs **and** *RNs* **compute** L_{ij} **received** from O_{ij}
  SGs **and** *RNs* **send** *message* with L_{ij} **and** U_{ij} to SG_i
  SG_i **selects** best L_{ij} + U_{ij}
  if *(SG_i **has** best L_{ij} + U_{ij})* **then**
    return
  end
  else
    SG_i **notifies** SG/RN with best L_{ij} + U_{ij}
    SG/RN.attach(O_{ij})
    SG_i.detach(O_{ij})
  end
end

6.3.2.2 Re-Clustering of Low-Performing Objects algorithm

The second REMOS-IoT algorithm, described in Algorithm 6.2, is the *Re-Clustering of Low-Performing Objects algorithm*. This algorithm finds better gateways for low-performing objects and matches them with the most relevant gateway (i.e. the one with most devices consuming/providing services to the object).

Devices not attached to their closest (according to the L_{ij} metric) and most relevant (R_{ij}) gateway are reconnected to a suitable gateway. Performance (P_{ij}) and age of information (A_{ij}) are other metrics combined in the final reputation score (U_{ij}) used for improving clusters. These scores are detailed in section 6.3.3.

The algorithm expects to attach devices to their most relevant gateway, when devices are not yet attached to them. The performance of the most relevant gateway (P_i) is also considered, as devices will not be re-attached if the most relevant gateway has poor performance.

In relation to devices performing poorly, based on their performance scores, the algorithm tries to attach these devices to better performing gateways, locating the next most relevant gateway if the current one is the most relevant.

Algorithm 6.2 Re-Clustering Low-Performing Objects

Require: P_{ij}, R_{ij}, A_{ij} and L_{ij} per O_{ij} in SG_i
foreach (O_{ij}) **do**
  $ITINP$ **computes** U_{ij}
  if (SG_i ***is not*** *max* L_{ij}) ***or*** (SG_i ***is not*** *most relevant SG* ***and*** *most relevant SG* P_i >= SG_i P_i) **then**
    most relevant SG.attach(O_{ij})
    SG_i.detach(O_{ij})
  end
  if (O_{ij} ***has*** *low* U_{ij} ***and*** *most relevant SG* = SG_i) **then**
    next most relevant SG.attach(O_{ij})
    SG_i.detach(O_{ij})
  end
end

Heuristic approaches were employed in the two algorithms, instead of analytical solutions due to the nature of the IoT device selection problem, which is NP-hard, as demonstrated in other works focused on IoT performance optimisation [239]–[242].

6.3.3 Reputation Score and Utility Functions

One of REMOS-IoT key features is the **reputation score (U_{ij})**. The score U_{ij} combines QoS, relevance, age of information and location scores of REMOS-IoT. The reputation score is employed in admission and performance control (which is achieved by improving device clusters connecting them to suitable gateways or relay nodes) and detection of low-performing objects. This score allows REMOS-IoT algorithms to create improved device clusters as it indicates which devices are performing poorly and are affecting the overall network performance. Regarding device mobility, current and neighbouring gateways and relay nodes exchange control messages and, based

on device reputation and location scores, the best gateway or relay node is selected to support moving devices communication.

Every service provided by a device has a performance score (P_{ijk}) as demonstrated in eq. (6.1). P_{ijk} is a normalisation of the sum of the normalised values of each metric M_x measured per service (a device can provide one or more services). These performance scores are averaged to compose the device **QoS score (P_{ij})**, as seen in eq. (6.2). Smart gateways maintain the scores of current and previously connected devices. The QoS score of a smart gateway (P_i) is also calculated using this formula.

$$P_{ijk} = \sum_{x} \left(\frac{M_x^{ijk}}{\max\limits_{x}(M_x^{ijk})} \right) \tag{6.1}$$

$$P_{ij} = \underset{k}{avg}(P_{ijk}) \tag{6.2}$$

The **Relevance score (R_{ij})** of a device O_{11} is an array of tuples formed by the ID of each smart gateway that has interacted with the device and the percentage of total packet communication exchanged between the gateway and the device. For example, an object has sent and received 200 packets from objects located in the same smart gateway (e.g. SG_1) and 800 packets from objects located in a different one (e.g. SG_2). Therefore, the device's relevance score is going to be R_{11}=[SG_1,20%],[SG_2,80%].

The **Age of Information score (A_{ij})** is a performance metric that indicates the freshness of information, and it is also included in the scores which compose the reputation score. The average age of information of a device is determined by a function for the M/M/1 queue model, given in [243] as seen in eq. (6.3). The arrival rate λ (which indicates the generation and submission of packets) is determined by a Poisson process, and average service time is given by $1/\mu$, where $\mu$is the service rate. The utilisation $\rho = \lambda/\mu$ varies according to each service provided by the devices.

$$A_{ij} = \frac{1}{\mu}\left(1 + \frac{1}{\rho} + \frac{\rho^2}{1-\rho}\right) \tag{6.3}$$

The **Location score (L_{ij})** of a device is based on its Relative Received Signal Strength (RSSI) or Reference Signal Received Power (RSRP) in relation to the neighbouring gateways or relay nodes (in D2D communications). REMOS-IoT uses the RSSI collected by gateways in Wi-Fi access points (AP) and RSRP collected by gateways in LTE base stations (BS) as the metrics for device location. These metrics were chosen because even simple IoT devices such as beacons, provide RSSI, as demonstrated in

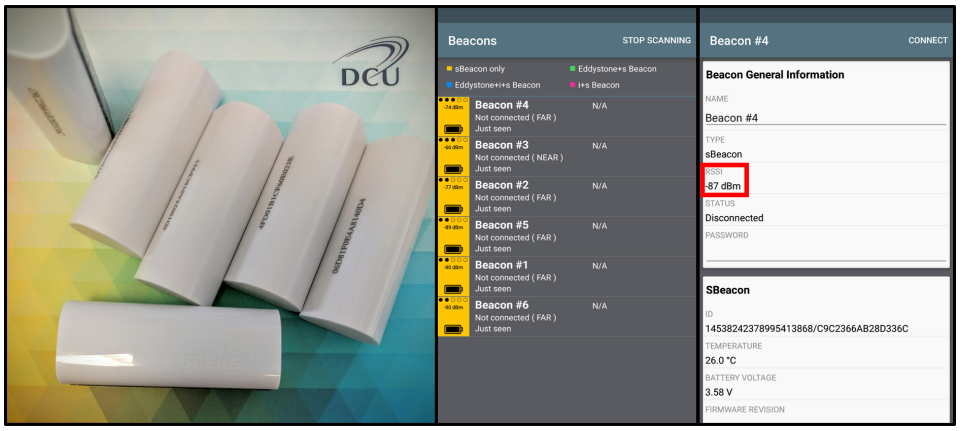


Figure 6.5: RSSI of Beacons Retrieved with the Beeks Beacon Maker App

Figure 6.5. In the area of distance estimation, RSSI-based methods are useful due to their low cost, low power and accessibility, being used in several types of systems.

REMOS-IoT considers -90 decibel-milliwatts (dBm) to be a low RSSI, according to the Table 6.1, adapted from [244]. REMOS-IoT takes decisions when devices present low dBm, which likely means that they are moving away from the gateway. This will be further detailed in the upcoming sections.

Table 6.1: RSSI Levels

RSSI	**Signal Strength**
$\geq$ -60 dBm	Excellent
-61 dBm to -75 dBm	Good
-76 dBm to -80 dBm	Fair
-81 dBm to -89 dBm	Poor
$\leq$ -90 dBm	Very Poor

The overall utility function used in the calculation of the reputation score is presented in eq. (6.4). REMOS-IoT computes the reputation of a device in relation to the smart gateways in order to find a balance between performance, age of information, relevance and location of devices, resulting in the normalised U_{ij} reputation score per object. The weights' sum must be equal to 1.

$$U_{ij} = wp\frac{P_{ij}}{\max\limits_{j}(P_{ij})} + wr\frac{R_{ij}}{\max\limits_{j}(R_{ij})} + wa\frac{A_{ij}}{\max\limits_{j}(A_{ij})} + wl\frac{|L_{ij}|}{\max\limits_{j}(|L_{ij}|)} \tag{6.4}$$

Table 6.2: Simulation Setup

Parameter	Value
Simulator	NS-3.24.1
Duration of the Simulation	15s+12s before and after sim.
Initial dist. between nodes and antennas	3 metres
WiFi and LTE Data Rates	40 Mbps and 100 Mbps
WiFi Standard	802.11ac (40MHz, MCS 9)
LTE eNB Antenna Model Type	Isotropic Antenna Model
Remote Station Manager	ConstantRateWifiManager
Mobility Model	ConstantVelocityMobilityModel
Speed of Smart Watch user	2 metres per second

The location score L_{ij} is used to consider device mobility (see Algorithm 6.1). The U_{ij} score reveals how devices are performing and enables to identify those which perform badly affecting cluster performance. They will be the object of the re-cluster algorithm (see Algorithm 6.2).

The research goal in this area is to find a solution that maximises the utility U_{ij}. However, an analytical solution cannot be found. REMOS-IoT implements a good, but not optimal, solution with the use of a heuristic mobility-enabled clustering approach, iteratively increasing performance of IoT devices, relay nodes and smart gateways. This can be applied in real time and has low overhead.

6.4 REMOS-IoT Performance Analysis

For testing purposes, REMOS-IoT was modelled and evaluated using the NS-3 simulator. The parameters used in the simulation setup including simulation duration, mobility speed, data rates, and mobility and antenna models and their values are presented in Table 6.2. Equal weights were applied to the reputation score (U_{ij}) used in REMOS-IoT algorithms (i.e. $wp = wr = wa = wl$). The reputation score was presented in section 6.3.3, equation 6.4.

Nine scenarios, as demonstrated in Table 6.3, were designed with clusters representing smart houses/businesses containing IoT objects. Mobile devices (exemplified as smart watches) are located at a smart home with other IoT objects all interconnected by a local gateway using a WiFi 802.11ac access point. These mobile devices move towards another smart home and when out of the reach of both gateways, it needs to connect to a relay node using D2D communications to continue online.

Table 6.3: REMOS-IoT Scenarios Description

11 devices in WiFi + 11 devices in LTE					
Amount of Devs.	Device Type	Scenario 1 (Mbps)	Scenario 2 (Mbps)	Scenario 3 (Mbps)	Traffic Direction
1X	High Bitrate	10	20	30	Downlink only
1X	Average Bitrate	5	10	15	Downlink only
1X	Low Bitrate	2	4	6	Downlink only
7X	Very Low Bitrate	0.4	0.8	1.2	Uplink and Downlink
1X	Moving Device	1	2	3	Uplink and Downlink
Total Bitrate		20.8	41.6	62.4	
22 devices in WiFi + 22 devices in LTE					
Amount of Devs.	Device Type	Scenario 4 (Mbps)	Scenario 5 (Mbps)	Scenario 6 (Mbps)	Traffic Direction
2X	High Bitrate	5	10	15	Downlink only
2X	Average Bitrate	2.5	5	7.5	Downlink only
2X	Low Bitrate	1	2	3	Downlink only
14X	Very Low Bitrate	0.2	0.4	0.6	Uplink and Downlink
2X	Moving Device	0.5	1	1.5	Uplink and Downlink
Total Bitrate		20.8	41.6	62.4	
100 devices in WiFi + 100 devices in LTE					
Amount of Devs.	Device Type	Scenario 7 (Mbps)	Scenario 8 (Mbps)	Scenario 9 (Mbps)	Traffic Direction
95X	Very Low Bitrate	0.208	0.416	0.624	Uplink and Downlink
5X	Moving Device	0.208	0.416	0.624	Uplink and Downlink
Total Bitrate		20.8	41.6	62.4	

Each gateway of the two smart homes, one connected to WiFi access points and one connected to an LTE base station, support a number of devices types in WiFi and LTE networks: three smart objects of high, average and low bitrates (e.g. in scenario 2, these device types have 20, 10 and 4 Mbps, respectively, representing video devices) with downlink only; and seven smart objects very low rates, simulating diverse mobile and non-mobile IoT objects. The mobile devices in the scenarios move from the WiFi LAN to the LTE network. When the device is in between networks, out of their covered area, an available relay node provides a D2D shared connection, so the smart watch is still under the ITINP platform.

Besides the multiple device types in each scenario, different amount of devices were tested: 22 devices in scenarios 1, 2 and 3; 44 devices in scenarios 4, 5 and 6; and 200 devices in scenarios 7, 8 and 9. Half of devices were located in the WiFi network and the other half in the LTE network. The mobile devices start in the WiFi network, moving towards the LTE network. In the scenarios with 200 devices (i.e 7, 8 and 9) only device types of very low bitrate were included, representing an IoT network with numerous constrained devices.

Three levels of total bitrate were shared among the different device types in each network. Scenarios 1, 4 and 7 have a low load of 20.8 Mbps when all bitrates are summed; scenarios 2, 5 and 8 have an average load 41.6 Mbps available for the devices; and scenarios 3, 6 and 9 have a high load of 62.4 Mbps divided among devices. The scenarios also include the other REMOS-IoT features, such as regular analyses of performance and objects' locations, re-clustering devices affecting network performance in order to increase quality of data delivery for the entire IoT network under ITINP's domain.

Each of the Figures A.1–A.54, presented in Appendix A, contains two plots to demonstrate network metrics before (a) and after (b) the smart watch moved from the WiFi network to LTE. In simulation time, the change of the smart watch from one gateway to another happened around 6.5s, therefore plots (a) x-axis cover the first 6.5s of the simulation, demonstrating devices performance before the mobile devices moved to the second network. Plots (b) demonstrate the new arrangement of devices, and the effects in performance from seconds 7s until 17s, as shown in the x-axis.

6.4.1 Analysis of Scenario 1

Scenario 1 consists of 11 devices connected to the WiFi network and 11 devices connected to the LTE network. The combined bitrate of devices is 20.8 Mbps, in each network, therefore, it is low load scenario, without much congestion. One device operates at a higher bitrate of 10 Mbps, with downlink only; the second device operates at an average bitrate of 5 Mbps, with downlink only; the third device has a lower bitrate of 2 Mbps, with downlink only; seven devices have a very low bitrate of 400 Kbps with both uplink and downlink communications; and one device is moving from the WiFi network to LTE, and it has a bitrate of 1 Mbps, receiving and sending data with downlink and uplink.

6.4.1.1 Downlink Analysis

Even though this scenario is not very congested, results for downlink indicated improvements in the selected metrics (i.e. delay, packet loss and throughput). After the mobile device moved from the first WLAN to the relay node, and then to the LTE network, devices throughput (Figure A.1) increased (i.e. the 10 Mbps, 5 Mbps, 2 Mbps devices had throughput increases of 3%, 3%, and 1.5%, respectively, while the seven remaining IoT objects had an average throughput increase of about 1%). Devices could, then, on average, achieve their maximum throughputs (e.g. The 10 Mbps device achieved a maximum throughput of 48% of the total available bitrate, which within the simulation it is the highest it could get). In terms of average packet loss, it was decreased in all devices to around 1% (Figure A.2). Average delay in the mobile device decreased from 120ms to 5ms, which represents an improvement of 96% (Figure A.3).

6.4.1.2 Uplink Analysis

Results for uplink demonstrate some gains, even though the uplink in this scenario was not very affected by congestion. Throughput had a small increase of 2% (Figure A.4), while average packet loss (Figure A.5) was decreased in 1%. The mobile device uplink delay was reduced to 7ms (Figure A.6). The devices of 10, 5 and 2 Mbps do not have uplink traffic.

6.4.2 Analysis of Scenario 2

Scenario 2 consists of 11 devices connected to the WiFi network and 11 devices connected to the LTE network. The combined bitrate of devices is 41.6 Mbps, in each network, which means it is a more congested scenario in comparison to scenario 1. One device operates at a higher bitrate of 20 Mbps, with downlink only; the second device operates at an average bitrate of 10 Mbps, with downlink only; the third device has a lower bitrate of 4 Mbps, with downlink only; seven devices have a very low bitrate of 800 Kbps with both uplink and downlink communications; and one device is moving from the WiFi network to LTE, and it has a bitrate of 2 Mbps, receiving and sending data with downlink and uplink.

6.4.2.1 Downlink Analysis

Results for downlink indicated significant improvements in delay, packet loss and throughput. After the mobile device (e.g. smart watch) moved from the first WLAN to the relay node, and then to the second network, devices throughput (Figure A.7) increased significantly (i.e. the 20 Mbps, 10 Mbps, 4 Mbps devices had throughput increases of 29%, 15%, and 6%, respectively, while the seven remaining IoT objects had an average throughput increase of about 5%). In terms of average packet loss, a decrease of 11% was observed, nearly eliminating it (Figure A.8). Improvements in delay were also noticed, decreasing from 133ms to 5ms (Figure A.9) in the mobile object. The IoT objects located in the first WLAN also experienced gains in relation to downlink performance. Average delay decreased from 132ms to 89ms in fixed objects, which represents an improvement of 32.5% and packet loss was reduced to 7% in these devices.

6.4.2.2 Uplink Analysis

Results for uplink demonstrate some gains, especially for the mobile device, as its throughput had a small increase of 2% (Figure A.10), while average packet loss (Figure A.11) had a modest decrease of 0.2%. The smart watch uplink delay was reduced to 3ms (Figure A.12). The remaining devices were not much impacted in terms of their average packet loss, delay and throughput, which remained roughly the same. The devices of 20, 10 and 4 Mbps do not have uplink traffic.

6.4.3 Analysis of Scenario 3

Scenario 3 contains 11 devices connected to the WiFi network and 11 devices connected to the LTE network. The combined bitrate of devices is 62.4 Mbps, in each network, a congested scenario. One device operates at a higher bitrate of 30 Mbps, with downlink only; the second device operates at an average bitrate of 15 Mbps, with downlink only; the third device has a lower bitrate of 6 Mbps, with downlink only; seven devices have a low bitrate of 1.2 Mbps with both uplink and downlink communications; and one device is moving from the WiFi network to LTE, and it has a bitrate of 3 Mbps, with downlink and uplink.

6.4.3.1 Downlink Analysis

Due to congestion, scenario 3 is more challenging. Results for downlink indicated improvements in delay, packet loss and throughput. After the mobile device moved from the first WLAN to the relay node, and then to the LTE network, devices throughput (Figure A.13) had an increase (i.e. 30 Mbps, 15 Mbps, 6 Mbps devices had throughput increases of 7%, 3%, and 0.5%, respectively, while the seven remaining IoT objects had an average throughput increase of about 7%). In terms of average packet loss, it was decreased in the devices in static position from 19.19% to 15.9%, and for the mobile device, packet loss was avoided, with a decrease of 10.3% (Figure A.14). Average delay in static devices decreased from 134ms to 38ms, and for the mobile device it decreased from 126ms to 5s (Figure A.15). Devices in the LTE gateway also had benefits, with delay decreasing in 5ms in fixed devices, loss slightly reduced, and throughput which was already high in the LTE devices, increased.

6.4.3.2 Uplink Analysis

The devices which have uplink (mobile device and devices with 3 Mbps) had their throughput increasing in up to 2% (Figure A.16), while average packet loss (Figure A.17) was decreased to less than 1%. The mobile device uplink delay was reduced from an average of 6.5ms to 2.3ms (Figure A.18).

6.4.4 Analysis of Scenario 4

Scenario 4 has a higher amount of devices: 22 devices connected to the WiFi network and 22 devices connected to the LTE network. The combined bitrate of devices is 20.8 Mbps, without much congestion, but with higher amount of devices in comparison to scenario 1. Two devices operate at a higher bitrate of 5 Mbps, with downlink only; two other devices represent the second device type, operating at an average bitrate of 2.5 Mbps, with downlink only; the third device type has a lower bitrate of 1 Mbps and there are two of such devices, with downlink only; 14 devices have a very low bitrate of 200 Kbps with both uplink and downlink communications; and two devices are moving from the WiFi network to LTE, and they have a bitrate of 500 Kbps, receiving and sending data with downlink and uplink.

6.4.4.1 Downlink Analysis

This scenario is not very congested, but contains a higher number of devices. After the mobile device moved from the first WLAN to the relay node, and then to the LTE network, devices throughput (Figure A.19) increased (i.e. the 5 Mbps, 2.5 Mbps, 1 Mbps devices had average throughput increases of 2%, 3%, and 1.4%, respectively, while the 14 remaining IoT objects had an average throughput increase of about 1%), achieving their maximum throughputs for most of the time. In terms of average packet loss, it was decreased in all devices, varying from 0.32% on the mobile device to 2.3% in the WiFi devices (Figure A.20). Average delay was mainly decreased in mobile device, changing from 123ms to 5s (Figure A.21).

6.4.4.2 Uplink Analysis

Regarding uplink, throughput had an increase of 2% (Figure A.22), while average packet loss (Figure A.23) was decreased reduced to less than 1% in all devices. The mobile device uplink delay was reduced from 11.2ms to 7.8ms, and also reduced in the other WiFi and LTE devices, from 5.18ms and 13.8ms to 4.2ms and 8.8ms, respectively (Figure A.24).

6.4.5 Analysis of Scenario 5

Scenario 5 contains 44 devices in total: 22 devices connected to the WiFi network and 22 devices connected to the LTE network, with a combined bitrate of devices is 41.6 Mbps. Two devices operate at a higher bitrate of 10 Mbps, with downlink only; third and fourth devices operate at an average bitrate of 5 Mbps, with downlink only; fifth and sixth devices have a lower bitrate of 2 Mbps, with downlink only; 14 devices have a very low bitrate of 400 Kbps with both uplink and downlink communications; and the remaining two devices are mobile and have a bitrate of 1 Mbps each, with downlink and uplink.

6.4.5.1 Downlink Analysis

Results for downlink indicated significant improvements after the mobile device moved from the first WLAN to the relay node, and then to the second network. Devices throughput (Figure A.25) increased significantly, with the 10 Mbps, 5 Mbps and 2 Mbps devices having throughput increases of 30%, 15%, and 6%, respectively. The 14 IoT objects with low bitrate had an average throughput increase of about 8%). Average packet loss was decreased from 21.8% to 18.2% (Figure A.26) in WiFi devices, slightly reduced in LTE devices (from 1% to 0.84%) and almost eliminated on mobile devices (falling from 11.5% to 0.19%). Delay decreased from 147ms to 70ms (Figure A.27) in the WiFi static devices, and continued low in LTE devices, maintaining the delay in 10ms even with the mobile devices being added to the LTE network. Delay also decreased from 140.9ms to 5ms in the mobile devices, after they moved in the network.

6.4.5.2 Uplink Analysis

Devices with uplink had a small increase of up to 1.5% (Figure A.28) in their uplink throughputs, while packet loss (Figure A.29) was almost completely avoided. The mobile devices uplink delay was reduced to 3.02ms (Figure A.30), while the WiFi devices and LTE fixed devices had their delays decreased to 1.25ms and 8.8, respectively.

6.4.6 Analysis of Scenario 6

Scenario 6 consists of 44 devices, 22 of these devices are connected to the WiFi network and 22 devices are connected to the LTE network. Their combined bitrates result in 62.4 Mbps, in each network. The two devices with higher load operate at 15 Mbps, with downlink only; two other devices operate at an average bitrate of 7.5 Mbps, with downlink only; two devices have a lower bitrate of 3 Mbps, with downlink only; 14 devices have a low bitrate of 600 Kbps with both uplink and downlink communications; two devices which are moving from the WiFi network to LTE, have a bitrate of 1.5 Mbps, with downlink and uplink.

6.4.6.1 Downlink Analysis

Scenario 6 is a congested scenario with a high amount of devices. After the mobile devices moved from the first WLAN to the relay node, and then to the LTE network, devices throughput (Figure A.31) had an increase (i.e. 15 Mbps, 7.5 Mbps, 3 Mbps devices had throughput increases of 4%, 2%, and 4%, respectively. The 14 remaining fixed IoT objects had an average throughput increase of about 6%). Packet loss decreased in the devices in static position from 75.7% to 26.09% in the WiFi network and from 2.22% to 1.65% in the LTE network. Packet loss also decreased in the mobile devices, going from 15.8% to 0.7% (Figure A.32). Average delay in static devices decreased from 264ms to 183ms in the WiFI network, and from 15ms to 10ms in LTE. The mobile devices' delay decreased from 265.1ms to 5s (Figure A.33).

6.4.6.2 Uplink Analysis

The devices which have uplink had an increase of up to 2% (Figure A.34) in throughput, while average packet loss (Figure A.35) was decreased to less than 1%. The mobile devices uplink delay was reduced from an average of 17ms to 4.4ms (Figure A.36). Delays of fixed devices in both WiFi and LTE devices were reduced to 2ms and 13.9ms respectively.

6.4.7 Analysis of Scenario 7

Scenario 7 contains 200 devices, with 100 of these devices connected to the WiFi network and 100 devices connected to the LTE network. All devices in this scenarios have uplink and downlink. Their combined bitrates in scenario 7 result in 20.8 Mbps, in each network. This scenario represents an IoT network with a high amount of constrained objects, performing at 208 Kbps. 5 of these devices are moving from WiFi to LTE.

6.4.7.1 Downlink Analysis

After the mobile devices moved from the first WLAN to the relay node, and then to the LTE network, devices throughput (Figure A.37) had an increase of about 12%, going from 72% to 84% of their maximum throughput in WiFi and from 84% to 95% in LTE. Mobile devices throughput was increased from 59.8% to 89.7%. In this scenario, as all devices have the same bitrate, average maximum throughput is 100%. Packet loss decreased in the devices in static position, from 9.64% to 1.91% in the WiFi network and from 1.7% to 0.6% in the LTE network. Packet loss also decreased in the mobile devices, going from 36.2% to 2.8% (Figure A.38). Average delay in static devices decreased from 45.3ms to 31.4ms in the WiFi network, and from 10ms to 7ms in LTE. The mobile devices' delay decreased from 31.27ms to 15s (Figure A.39).

6.4.7.2 Uplink Analysis

Uplink in fixed devices had an increase of 4% in WiFi throughput and 7% in LTE, reaching 84.8% and 91.4% of devices maximum throughput, respectively (Figure A.40). Mobile devices had a throughput increase of 11%, from 79.7% to 90.1%. Average packet loss (Figure A.41) was decreased to about 1% in devices (0.77% in LTE devices, 1.08% in WiFi devices and 1.77% in mobile devices). The mobile devices uplink delay was reduced from an average of 20.38ms to 19.94ms (Figure A.42). Delays of fixed devices in both WiFi and LTE devices were reduced from 21.56ms to 13.24ms and from 27.97ms to 17.96ms, respectively.

6.4.8 Analysis of Scenario 8

Scenario 8 contains 200 devices: 100 devices connected to WiFi and 100 connected to LTE, and all devices have uplink and downlink and perform at 416 Kbps. Their combined bitrates result in 41.6 Mbps, in each network.

6.4.8.1 Downlink Analysis

After the 5 mobile devices moved from WiFi to LTE, devices throughput (Figure A.37) had an increase of about 26%, going from 42.9% to 68.7% of their maximum throughput in WiFi and from 67% to 78% in LTE. Mobile devices throughput was increased from 64.2% to 94.1%. Packet loss decreased in the devices in static position, from 26.18% to 17.1% in the WiFi network and from 1.24% to 0.44% in the LTE network. Packet loss also decreased in the mobile devices, going from 38.7% to 2.4% (Figure A.38). Average delay in static devices decreased from 75.13ms to 32.68ms in the WiFi network, and from 10ms to 7ms in LTE. The mobile devices' delay decreased from 39.49ms to 15s (Figure A.39).

6.4.8.2 Uplink Analysis

Uplink throughput increased 16% in WiFi and 9% in LTE, reaching 76.3% and 79.9% of devices maximum throughput, respectively (Figure A.40). Mobile devices had a throughput increase of 23%, from 64.27% to 90.1%. Average packet loss (Figure A.41) was decreased from 15.7% to 7.1% in WiFi devices, from 1.95% to 0.69% in LTE devices and from 14.26% to 1.67% in mobile devices. The mobile devices uplink delay was reduced from an average of 15.01ms to 10.63ms (Figure A.42). Delays of fixed devices in both WiFi and LTE devices were reduced from 13.62ms to 13.41ms and from 28.77ms to 18.55ms, respectively.

6.4.9 Analysis of Scenario 9

Scenario 9 also contains 200 devices: 100 devices connected to WiFi and 100 connected to LTE, and all devices have uplink and downlink and perform at 624 Kbps. Their combined bitrates result in 62.4 Mbps, in each network. This is the most challenging scenario, with high number of devices and bitrate requirements.

6.4.9.1 Downlink Analysis

Devices throughput (Figure A.49) in WiFi had an increase of about 12%, after the 5 mobile devices moved from WiFi to LTE, going from 42.7% to 50.6% of their maximum throughput and from 60.5% to 69.7% in LTE. Mobile devices throughput was increased from 41.6% to 75.9%. Packet loss decreased in the devices in static position, from 93.12% to 31.4% in the WiFi network and from 16.15% to 10.4% in the LTE network. Packet loss also decreased in the mobile devices, going from 68.02% to 8.39% (Figure A.50). Average delay in static devices decreased from 71.95ms to 31.58ms in the WiFi network, and from 10.59ms to 7.79ms in LTE. The mobile devices' delay decreased from 31.10ms to 15.41s (Figure A.51).

6.4.9.2 Uplink Analysis

Uplink throughput increased 4% in WiFi and 3% in LTE, reaching 51.3% and 70.6% of devices maximum throughput, respectively (Figure A.52). Mobile devices had a throughput increase of 30%, from 59.6% to 89.1%. Average packet loss (Figure A.53) was decreased from 38.24% to 26.49% in WiFi devices, from 18.61% to 10.38% in LTE devices and from 26.48% to 8.96% in mobile devices. The mobile devices uplink delay was reduced from an average of 57.55ms to 55.13ms (Figure A.54). Delays of fixed devices in both WiFi and LTE devices were reduced from 6.64ms to 6.48ms and from 98.02ms to 74.9ms, respectively.

6.4.10 Scenarios Overall Results

The plots from Figures 6.6–6.29 present comparisons between the nine scenarios. The red bars show results from the seconds 2 until 6.5, before mobile devices moved into the LTE network. The blue bars represent seconds 6.6 until the end of simulation in second 17, after mobile devices moved and REMOS-IoT algorithms were performed for relay and mobility support, and analysis was performed for downlink and uplink traffic.

With a continued connection for mobile devices, there is a seamless change from one network to another. High throughputs being achieved in most scenarios, in both uplink and downlink. Scenario 9, which contains a high number of devices and high bitrate requirements (i.e. 200 devices and 62.4 Mbps total bitrate), had less benefits in relation to throughput than other scenarios, due to the amount of congestion. Still, mobile devices, for instance, had an increase of 83% (Figure 6.14) of achieved throughput in this scenario.

Delays in both uplink and downlink scenarios in WiFi and LTE were greatly decreased. In scenario 6, for instance, a high delay of 265.1ms was decreased to 5ms for moving devices. Devices in WiFi, LTE and mobile ones had an average decrease of 43%, 17% and 92%, respectively, in delay across all scenarios for downlink. For uplink, the decreases in delay were 27%, 26%, and 23% for WiFi, LTE and mobile devices.

Packet loss was also greatly reduced across all scenarios. On average in downlink, packet loss was decreased in 51%, 36% and 92% for WiFi, LTE and mobile devices, when considering all scenarios. In relation to uplink, loss was decreased in 44%, 51%, and 80% in WiFi, LTE and mobile devices.

The obtained results from the nine scenarios show that REMOS-IoT is able to perform significant improvements in terms of throughput, packet loss and delay, the selected metrics for the study.

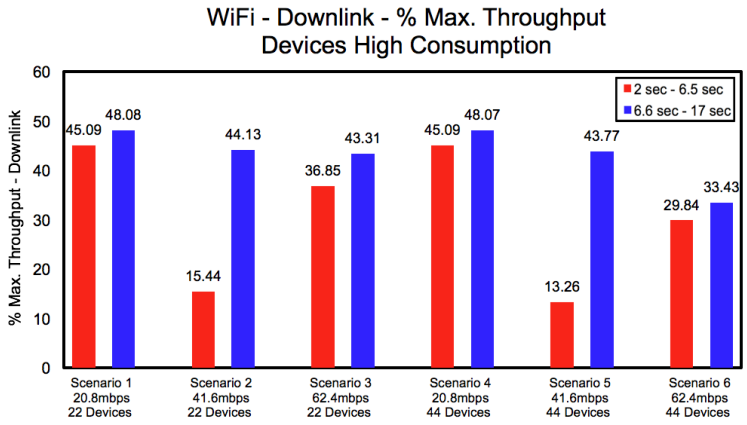


Figure 6.6: WiFi - Downlink - % Max. Throughput of Devices with High Consumption

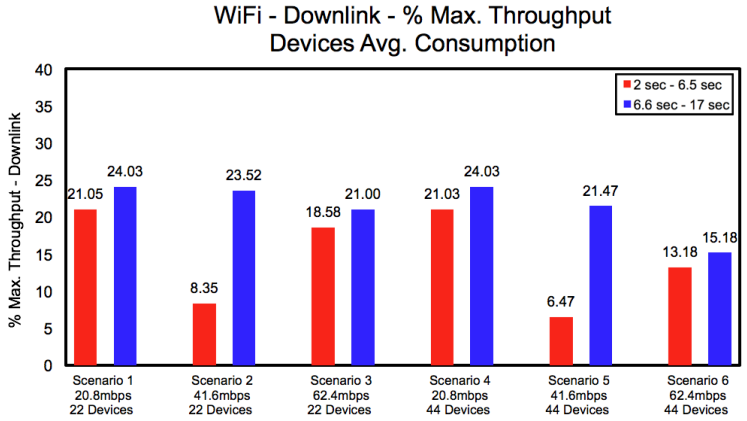


Figure 6.7: WiFi - Downlink - % Max. Throughput of Devices with Average Consumption

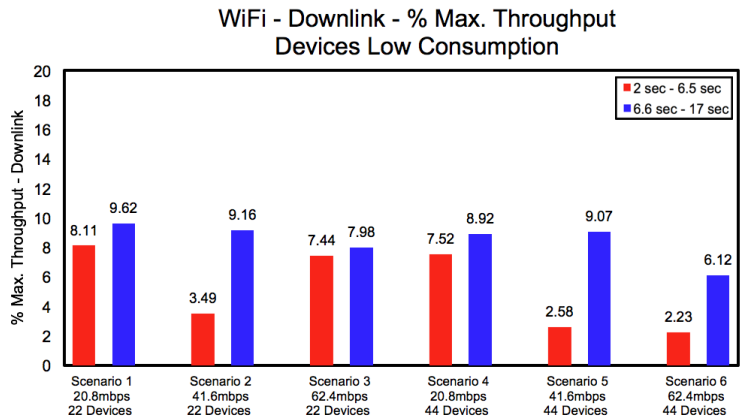


Figure 6.8: WiFi - Downlink - % Max. Throughput of Devices with Low Consumption

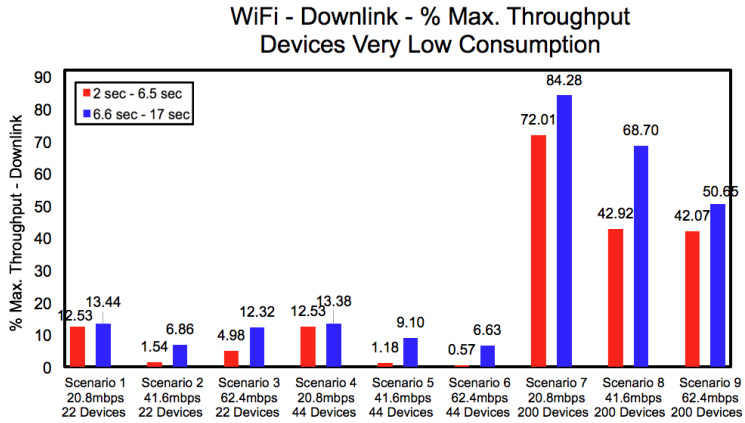


Figure 6.9: WiFi - Downlink - % Max. Throughput of Devices with Very Low Consumption

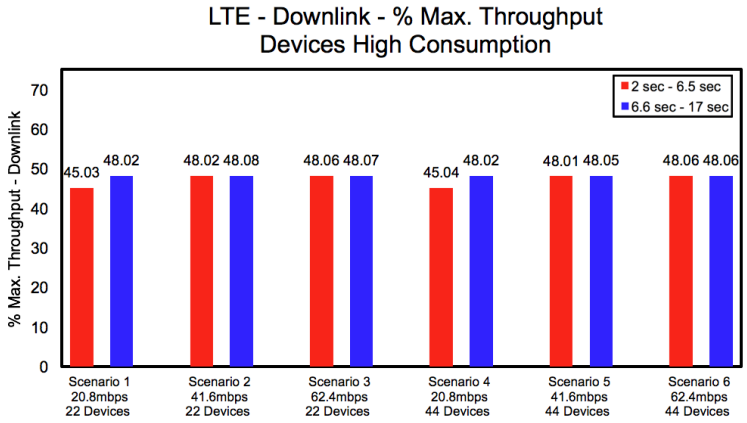


Figure 6.10: LTE - Downlink - % Max. Throughput of Devices with High Consumption

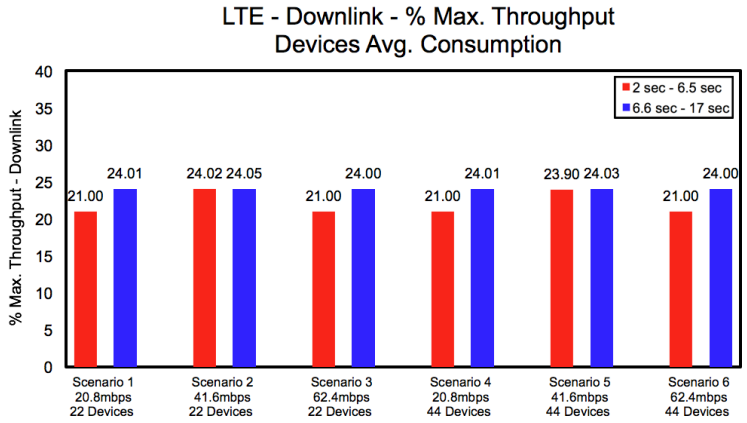


Figure 6.11: LTE - Downlink - % Max. Throughput of Devices with Average Consumption

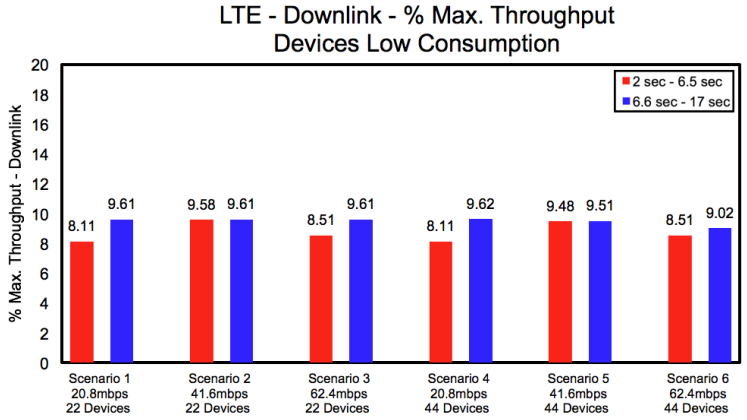


Figure 6.12: LTE - Downlink - % Max. Throughput of Devices with Low Consumption

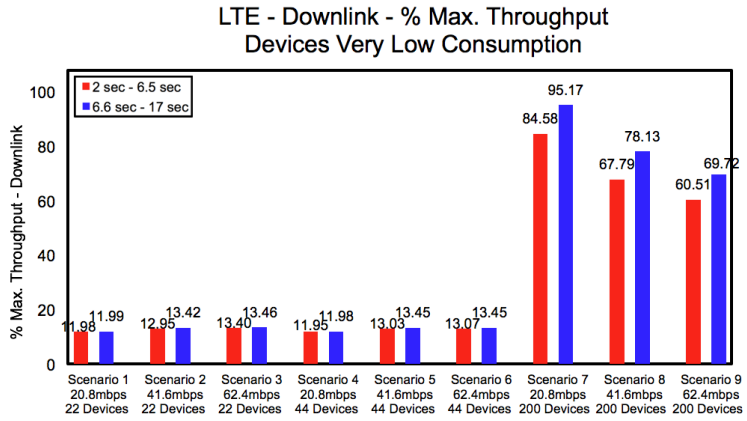


Figure 6.13: LTE - Downlink - % Max. Throughput of Devices with Very Low Consumption

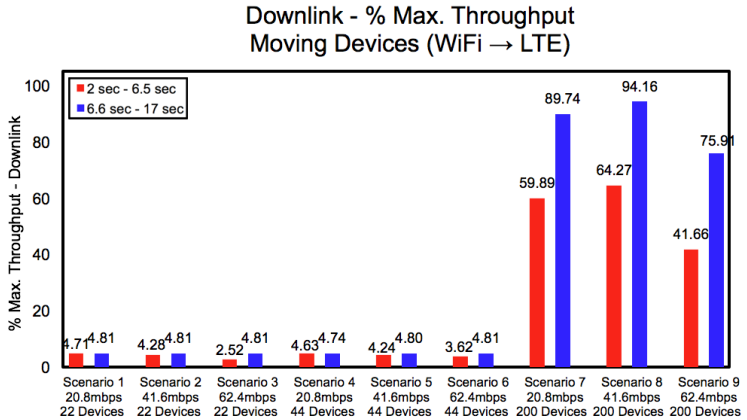


Figure 6.14: Downlink - % Max. Throughput of Devices Moving from WiFi to LTE

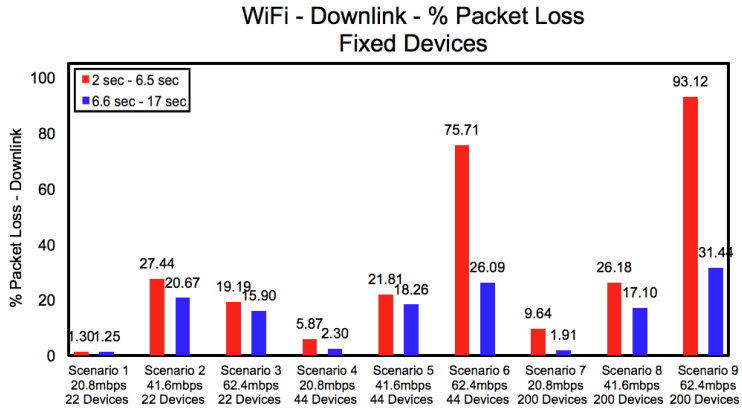


Figure 6.15: WiFi - Downlink - % Average Packet Loss of Fixed Devices

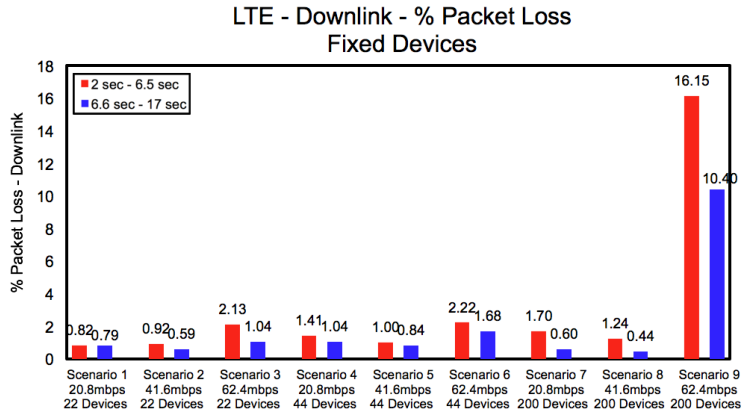


Figure 6.16: LTE - Downlink - % Average Packet Loss of Fixed Devices

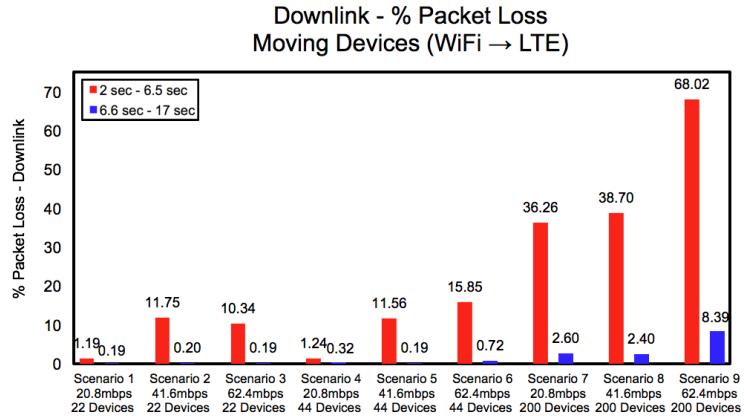


Figure 6.17: Downlink - % Average Packet Loss of Devices Moving from WiFi to LTE

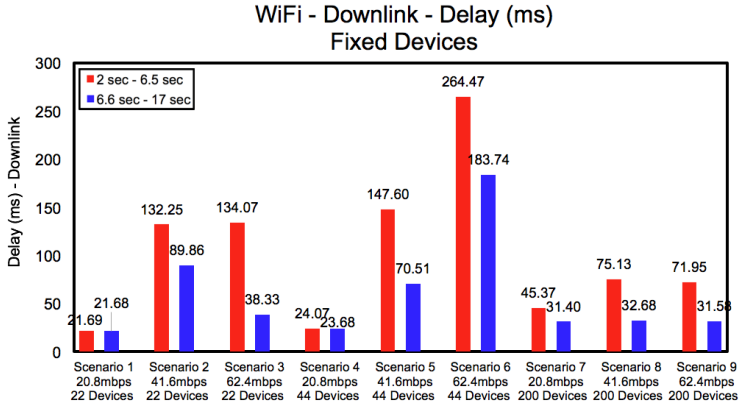


Figure 6.18: WiFi - Downlink - Average Delay (ms) of Fixed Devices

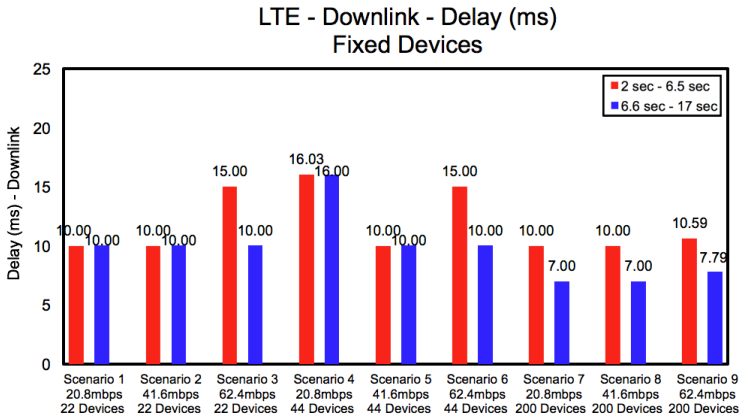


Figure 6.19: LTE - Downlink - Average Delay (ms) of Fixed Devices

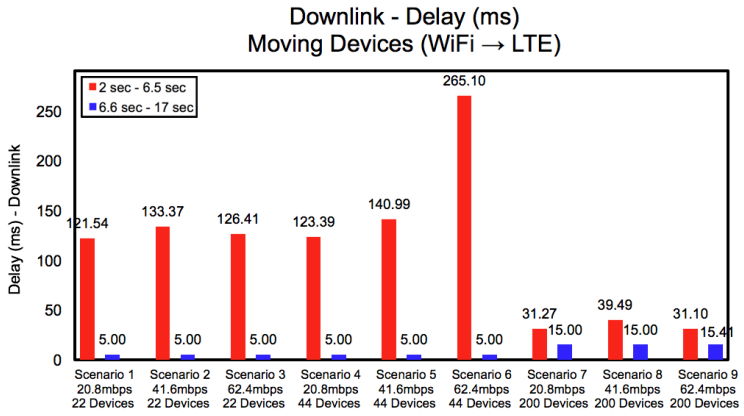


Figure 6.20: Downlink - Average Delay (ms) of Devices Moving from WiFi to LTE

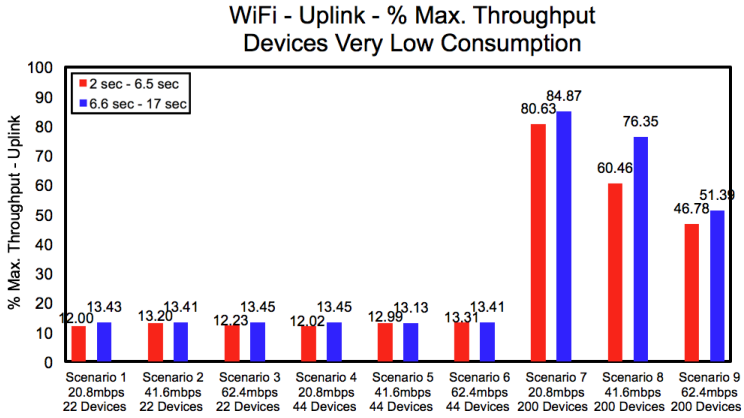


Figure 6.21: WiFi - Uplink - % Max. Throughput of Devices with Very Low Consumption

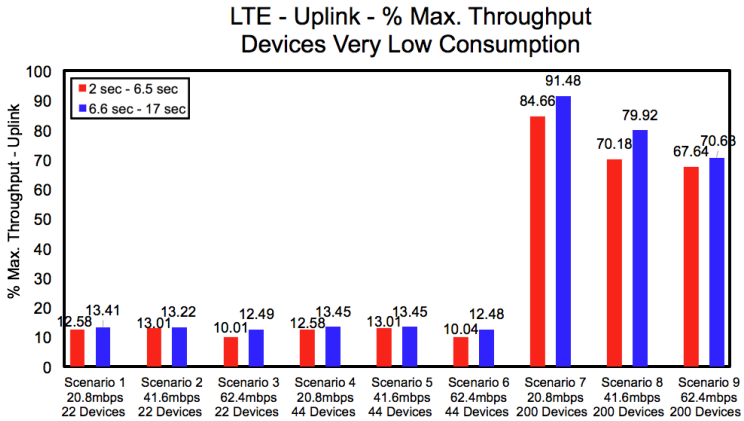


Figure 6.22: LTE - Uplink - % Max. Throughput of Devices with Very Low Consumption

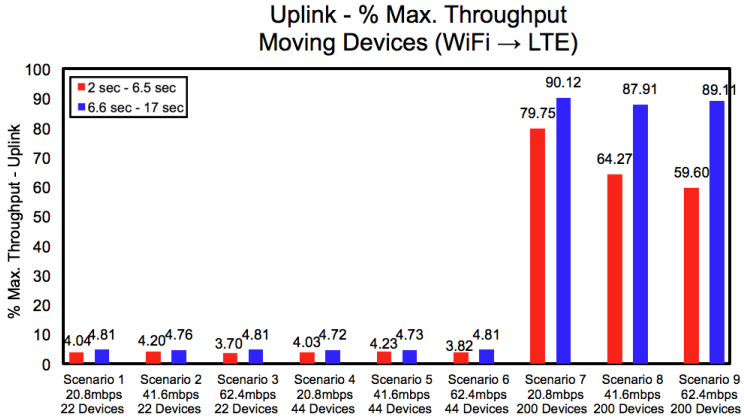


Figure 6.23: Uplink - % Max. Throughput of Devices Moving from WiFi to LTE

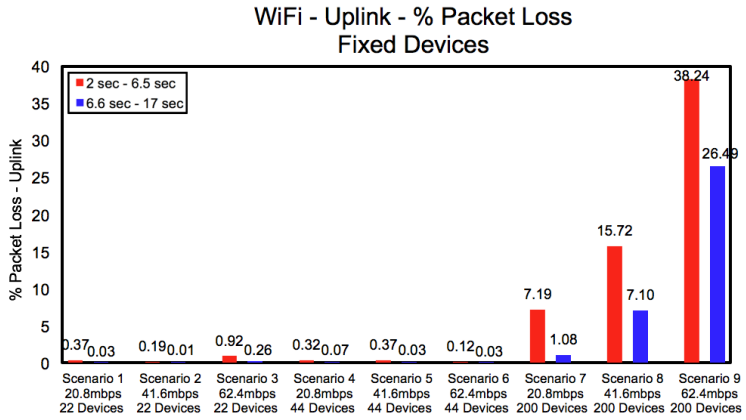


Figure 6.24: WiFi - Uplink - % Average Packet Loss of Fixed Devices

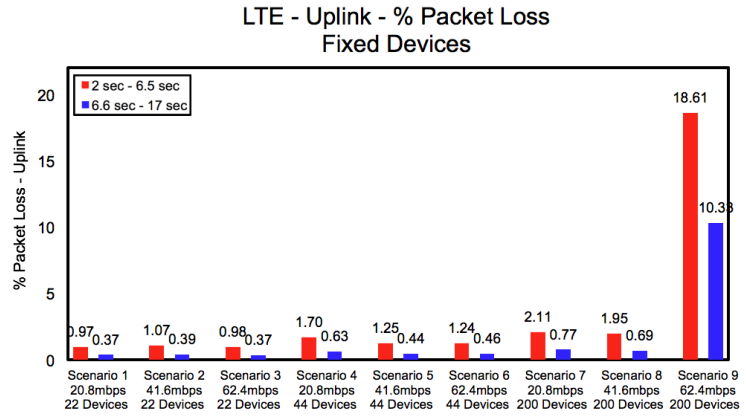


Figure 6.25: LTE - Uplink - % Average Packet Loss of Fixed Devices

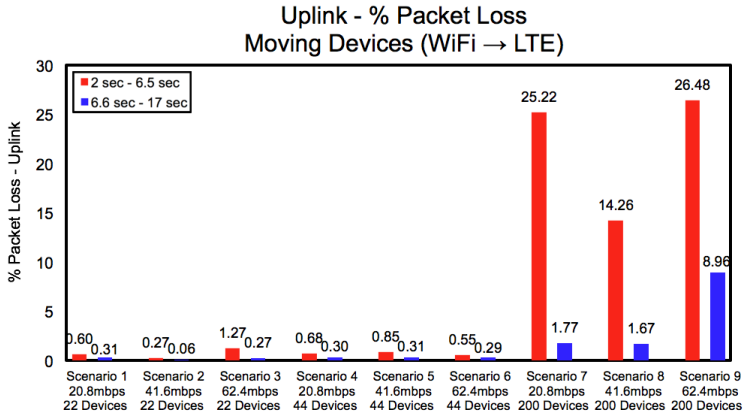


Figure 6.26: Uplink - % Average Packet Loss of Devices Moving from WiFi to LTE

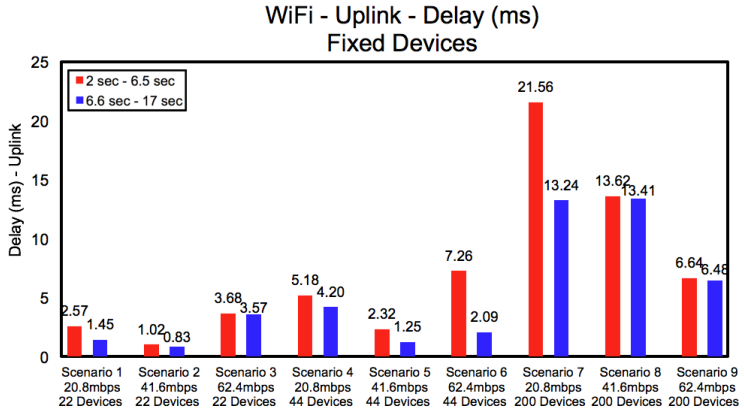


Figure 6.27: WiFi - Uplink - Average Delay (ms) of Fixed Devices

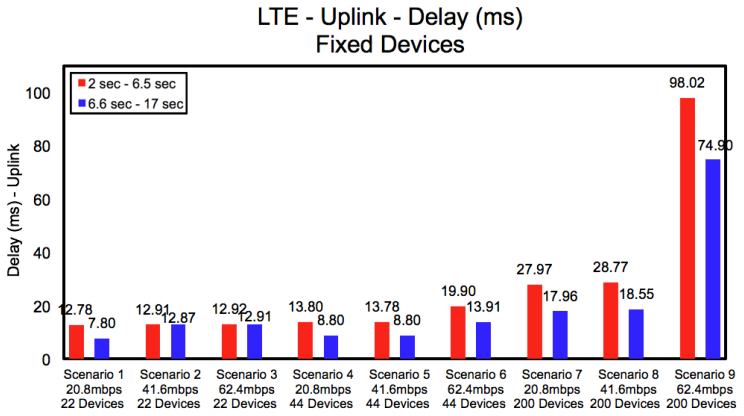


Figure 6.28: LTE - Uplink - Average Delay (ms) of Fixed Devices

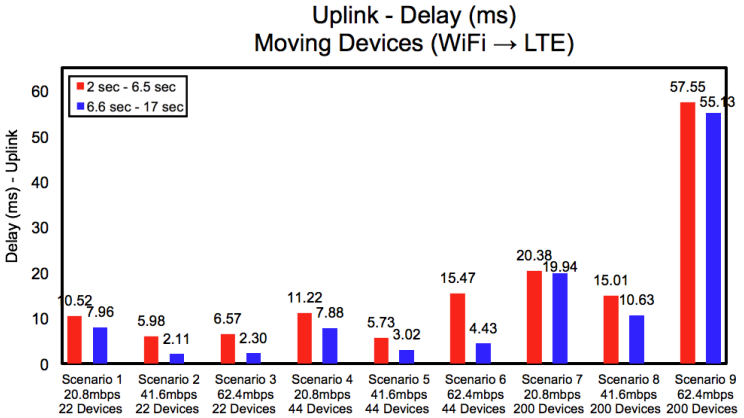


Figure 6.29: Uplink - Average Delay (ms) of Devices Moving from WiFi to LTE

Table 6.4: Baseline Comparison

	Delay (ms)		Loss (%)	
	IoT - RTP & IoT - RTCP	**REMOS-IoT**	**IoT - RTP & IoT - RTCP**	**REMOS -IoT**
Average	204.74	7.53	2.50	0.43
St. Dev.	2.83	0.11	1.34	0.08
Max. Value	211.00	7.75	5.00	0.60
Min. Value	200.00	7.25	1.00	0.35
T-test p-value	6.9×10^{-35} (<0.05)		3.25×10^{-6} (<0.05)	

6.4.11 Baseline Comparison

REMOS-IoT is compared against the IoT-RTP and IoT-RTCP adaptive protocols for multimedia transmission in IoT environments [245], in order to verify its gains in comparison with a recent IoT networking approach. IoT-RTP and IoT-RTCP are adaptive versions of the real-time transport protocol (RTP) and real-time control protocol (RTCP).

The IoT-RTP and IoT-RTCP protocols, deployed on the Network Simulator 2 (NS-2), employ a novel mechanism of dividing large multimedia sessions into simpler sessions with awareness of network status. Although REMOS-IoT was deployed on NS-3, significant differences were not found when comparing the schemes, in relation to the simulator used, with results being alike. For the baseline comparison, delays and packet losses were averaged over time in all devices in the second WLAN, after REMOS-IoT algorithms were applied. A similar simulation runtime was considered. The obtained results are available in Table 6.4. REMOS-IoT outperforms the other schemes with 96.3% lower delay and 82.6% less packet loss, considering the average load scenario 2 of REMOS-IoT. Statistical significance in favour of REMOS-IoT, in both delay and loss samples, is evidenced by the low p-value of 6.9×10^{-35} for delay and 3.25×10^{-6} for packet loss, obtained following a paired student t-test result analysis.

6.5 Chapter Summary

This chapter detailed REMOS-IoT, which is employed on a D2D-enabled architecture for IoT devices. REMOS-IoT novelties include algorithms and the ability to record scores for QoS, relevance, age of information and location, in order to cluster IoT

devices efficiently and support mobility.

The testing scenario consists of mobile devices in a smart house with several other smart devices initially connected to a local gateway using a WiFi 802.11ac access point. The mobile devices loses connection and need to relay node before reconnection, which triggers REMOS-IoT algorithms.

In order to demonstrate REMOS-IoT benefits, the solution was tested via NS-3 modelling and simulations and results for uplink and downlink were recorded in terms of throughput, packet loss and delay, which were improved considerably. The proposed solution also outperforms another solution in terms of packet loss and delay.

Chapter 7

VRITESS - Virtual Reality-Internet of Things Environment Synchronisation Scheme for Social IoT

The last two chapters presented solutions that improve IoT devices performance, however, besides smooth performance of the IoT network, it is important that users understand and operate IoT devices correctly. VR is currently being used in many different areas such as car prototyping, gaming, medical training, teaching, etc. IoT devices such as systems-on-a-chip (e.g. Raspberry Pi), smart appliances and sensors support a wide range of services, including machine automation, remote monitoring and control. This chapter introduces a novel social VR-IoT environment, which allows users to share and control local or remote IoT devices in a user-friendly virtual platform integrated to ITINP, which was presented in the previous chapters. The virtualisation of devices also allows social networks to be integrated into the IoT network. Two approaches using the VR-IoT solution are presented: one local network-based and one cloud-based. The proposed VR-IoT environment contains VRITESS, the novel VR-IoT Environment Synchronisation Scheme [246], [247], which facilitates a consistent and integrated experience for users by enabling control of real IoT objects with VR headsets. Control of some IoT objects in extreme environments or devices which are complex to operate, can be simplified in a virtual environment. The VRITESS synchronisation scheme maintains the real objects updated, following instructions given in the virtual world and vice-versa. Testing involved local network-based and cloud-based testbeds created with a VR headset and IoT devices. The social IoT aspects of the solution are also highlighted with the design and implementation of a Twitter-based social IoT network integrated to the virtual environment [112].

7.1 Problem Statement

Devices such as smart appliances, wearables, health monitors, smart cars, etc. are changing the way users interact with devices. However, certain IoT devices are still complex to operate and do not provide a simple and mature user experience [7], [248]. Multimedia solutions such as [249]–[251], enable users to enjoy a rich media experience. These solutions combined with Virtual Reality (VR) technology can expand the perception of reality thanks to the introduction of realistic scenarios with auditory, tactile and visual capabilities, which translate the real world into an immersive virtual world [252], [253]. According to [8], it can be easier to interact and understand IoT devices by using gestures, language and other human senses. IoT objects such as landslide sensors or devices for water level monitoring, which can be located in hazardous places, and other IoT objects, which can be too complex to use, can be operated in an easier way in a virtual environment. The virtual environment also enables the social IoT to become reality: users will be able to consume content and share access to services and devices.

In order to simplify the use of IoT devices, the VR-IoT Environment Synchronisation Scheme (VRITESS) was built as an innovative mechanism, which enables users to seamlessly operate real IoT devices from a virtual environment, and vice-versa. Through VRITESS, users can visualise data provided by beacons and operate sensors and other features available in IoT devices, such as single-board computers (e.g. Raspberry Pi). The visualisation is performed in a VR environment with a user-friendly interface facilitating the interaction with real objects. When a user manipulates a device in the virtual environment, besides the actions being executed in the virtual world, they will also be executed on the real objects. VRITESS enables bridging real and virtual worlds, therefore, operations in the real objects will also be reflected in the virtual ones, with VRITESS being responsible for the synchronisation between real objects and virtual objects. Figure 7.1 illustrates the concept of interconnecting the IoT objects *ij*, which are networked by smart gateways *i*, to the VR platform, where they can be visualised and manipulated. The details of the IoT and VR platform integration will be discussed in the next sections.

VRITESS was deployed and tested in two different situations using the VR-IoT solution: a local network-based approach and a cloud-based approach, allowing the analysis of performance metrics, such as network latency. Both testbeds contain beacons, an Oculus Rift and Raspberry Pis. Tests were conducted in the Perform-

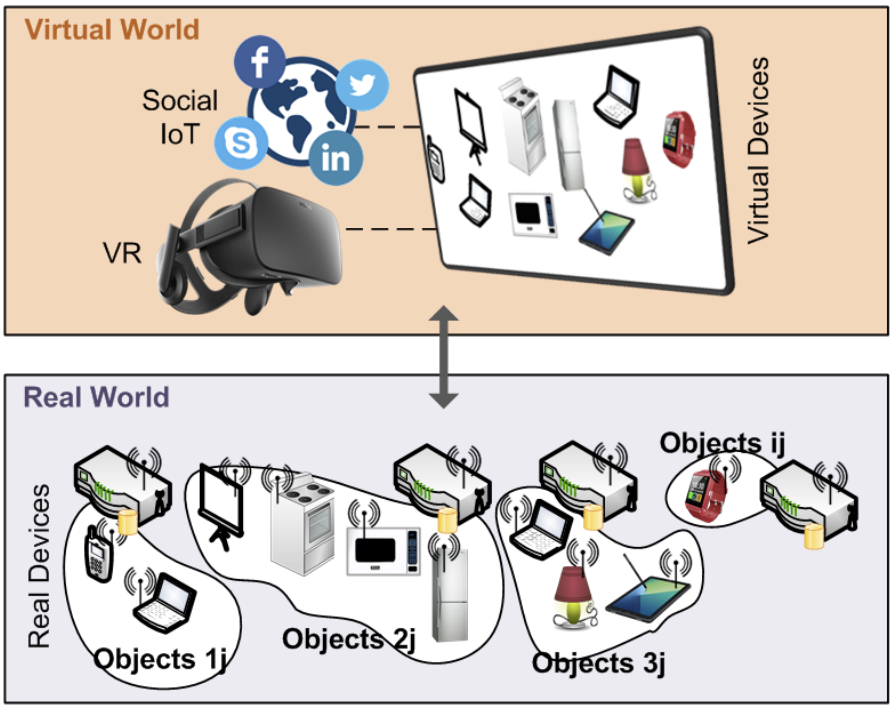


Figure 7.1: Real-world and virtual-world devices

ance Engineering Laboratory at the Dublin City University, Ireland. The VR environment recreated the laboratory in 3D, with virtual devices matching the real IoT ones located in the office. A synchronisation algorithm, which is part of the VRITESS solution, maintains the devices up-to-date, sending actions and events that happened in the virtual environment to the real objects, and vice-versa, updating changes of their states and maintaining consistency. Comparative testing results show that the cloud-based solution has slightly increased latency in comparison to the local-based deployment.

7.2 Design of the VR-IoT Synchronisation and VR-IoT Platform

This section introduces the the synchronisation solution and the VR platform, its components and the communication process among them.

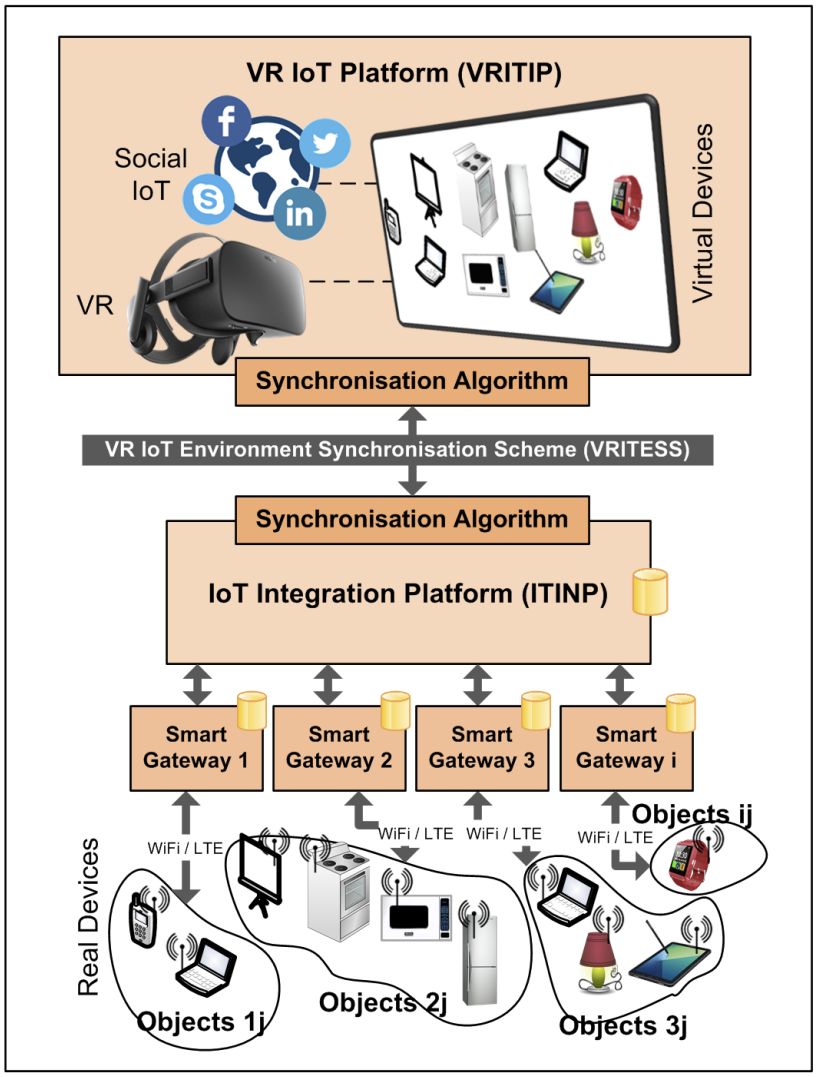


Figure 7.2: The VRITESS architecture

7.2.1 Architecture Description

VRITESS is implemented on top of an IoT architecture, as illustrated in Figure 7.2 and also presented in chapter 4. The architecture, which deploys VRITESS, has the following major components: *IoT objects* (e.g. smart appliances, wearables, health monitors, sensors) providing services to users and other devices – each of these IoT objects have two instances: real (the real IoT device) and virtual (the VR representation of the real object); *Smart Gateways* networking the IoT devices, allowing communication among connected objects; the *IoT Integration Platform (ITINP)*, a platform that contains a cloud-based server for the integration of services, including the interconnection of the smart gateways, and the *VR IoT Platform (VRITIP)*, composed of a VR

server and VR headsets for the rendering of 3D virtual spaces containing the virtual representation of IoT objects, allowing users to interact with them using straightforward controls. VRITIP also maintains virtual objects up-to-date with the operations executed in the real ones, showing them in the virtual environment.

VRITIP and ITINP communicate over local networks or the cloud, using IoT protocols such as the MQTT protocol. In the local network approach, a local database stores the operations executed in the virtual and real devices, and based on timestamps, it keeps both types of devices updated. In the cloud approach, a cloud-based IoT protocol-enabled database keeps the operations and timestamps, while both ITINP and VRITIP constantly access the data. VRITESS is implemented in both ITINP and VRITIP, keeping devices updated with the information stored in the local or cloud-based databases. Network Time Protocol (NTP) is used to synchronise ITINP and VRITIP clocks over the networks, avoiding synchronisation errors [254]. Details on the design and implementation of the concept and testbeds are presented in the upcoming sections.

7.2.2 Communication Process

A status structure is used by the gateways to store the actions regarding the devices. ITINP receives operations executed in real devices, VRITIP receives operations executed in virtual devices and VRITESS enables the two-way communication and synchronisation between these two. Both virtual and real devices need to be able to update their functionalities based on the latest information in these status structures.

Several devices can benefit from this communication structure, with operations such as turning on/off devices and appliances, reading sensor temperatures, movements and activities, turning on/off motors for opening and closing doors and curtains, etc. Users exist on both virtual and real world, so the status structure and its associated update mechanisms support tracking the way users interact with the devices in either world. The device status structure used in the local and cloud-based databases are presented in the following example (illustrating a virtual user turning off a virtual object):

```
{
"timestamp":                                    "2019-04-15T09:15:12.147"
"last_change":                                  "2019-04-15T09:14:23.133"
"user_id":                                                  "andersonsimis"
```

```
"type_of_user":                                          "virtual"
"type_of_device":                                        "virtual"
"tags:                                                ["POWER_INSTRUCTION"]
"device_id":                                         "0036:5E25:0000:21DC"
"data"                                                        "OFF"
}
```

This operation will result in the real device also turning off and the appropriate status structure update:

```
{
"timestamp":                                    "2019-04-15T09:15:12.235"
"last_change":                                  "2019-04-15T09:14:23.224"
"user_id":                                              "andersonsimis"
"type_of_user":                                               "virtual"
"type_of_device":                                                "real"
"tags:                                                ["POWER_INSTRUCTION"]
"device_id":                                         "0036:5E25:0000:21DD"
"data"                                                        "OFF"
}
```

This example illustrates a virtual user that turns off a virtual object, resulting in the same operation being executed on the real object. Operations performed in the virtual environment (e.g. turn on a device, change the temperature of a thermostat, etc.) are synchronised in ITINP and then updated in the database of the smart gateway responsible for interconnecting the corresponding real device, passing the instruction to the object using an IoT protocol such as MQTT.

The solution architecture contains databases at the level of the gateways and in the cloud, so it is possible to record the actions in real and virtual devices. When an action is performed, details are registered following the status structure presented in this section. Section 7.4 describes in detail the implementation of a local database (MySQL) in the local network-based solution and a cloud database (Adafruit IO) [255] in the cloud-based solution.

The database implementation also allows users to be identified when performing actions, therefore, when logging into the applications, users can only control the

devices that are shared with or owned by them. The social aspect of the platform is the possibility of granting access to other users to control devices in the virtual platform.

The VRITESS synchronisation algorithm keeps the devices updated, according to the operations executed in the virtual and real versions of the devices, respectively.

7.3 VRITESS Synchronisation Algorithm

The VRITESS synchronisation algorithm is employed on both real world devices, at the level of ITINP (acting on the real objects), and virtual world devices, at the level of VRITIP (acting on the virtual objects), in order to maintain consistency.

Algorithm 7.1 Synchronisation of real and virtual-world IoT Objects

Require: *old_t* ← The last timestamp sent by ITINP and VRITIP; *data[]* ← The set of data, following the structure in section III.B
Output : *new_t* ← New timestamp sent back to ITINP and VRITIP; *updated_data[]* ← Updated data sent back to ITINP and VRITIP

```
new_t = old_t
foreach data in data[] do
    if (data.timestamp > old_t) then
        updated_data[].add(data.value)
        new_t = max(data.timestamp,new_t)
    end
end
return new_t, updated_data[]
```

Algorithm 7.1 presents the synchronisation of real-world and virtual-world IoT objects. It works by sending to ITINP and VRITIP the latest action performed by real and virtual users, based on the timestamp value of these actions. ITINP and VRITIP employ the Network Time Protocol (NTP), therefore, their clocks are synchronised over the network.

As seen in algorithm 7.1, the timestamps and data structure maintain the real and virtual devices up-to-date and synchronised with each other. The communication process follows the structure introduced in Section 7.2.2. The *new_t* variable stores the most recent timestamp when new actions are received (i.e. newer than the *old_t* variable, which stores the last executed action). Some of the instructions may be informative (e.g. a virtual gauge representing a sensor temperature) or actions (e.g. turn on/off a virtual appliance resulting in the same action on the real appliance).

All the data sent by ITINP and VRITIP is organised into an array of objects containing all *data* to be processed by the algorithm. Each object of the array contains the structure for communication, with fields including: data.timestamp (current timestamp), data.last_change (timestamp of last data modification), data.device_id and data.user_id (unique IDs for device and user, respectively – strings used as keys), and other strings such as data.type_of_user, data.type_of_device, data.tags (indicating the type of action, e.g. power related or the sensor type) and data.data (indicating the action to be executed, e.g. turn off/on).

The algorithm is triggered on an event-based fashion, therefore, every new event, e.g. turning on a light, will call a function that inserts it into a database. After this insertion process in the local or cloud database, the synchronisation mechanism is triggered to send the latest action to the corresponding virtual or real device, if a real or virtual device was manipulated, respectively.

7.4 VRITESS Performance Analysis

The VRITESS solution was deployed in two different approaches in the Performance Engineering Laboratory at the Dublin City University, Ireland.

Both a local network-based solution and a cloud-based solution were created, so that comparative performance differences between the two approaches can be assessed.

In the network-based approach, a browser-based VR application was created, while in the cloud-based approach a 3D application was developed with Unity, a 3D engine. Both were tested with an Oculus Rift, real IoT devices and a Raspberry Pi, which interconnects the IoT devices using its General Purpose Input Output (GPIO) pins to control LEDs and a servo-motor, and Bluetooth to receive data from beacons.

7.4.1 Local Network-Based Approach

The local network-based implementation only allows local devices to be visualised in the VR headset. The VR headset and the computer that powers it communicate directly to the IoT devices using a local wireless network.

The testbed illustrated in Figure 7.3 has the following major components: an Oculus Rift [256], two Raspberry Pis [257] and four Beeks IoT beacons [258], which are connected to the Raspberry Pis via Bluetooth Low Energy. A Dell Alienware com-

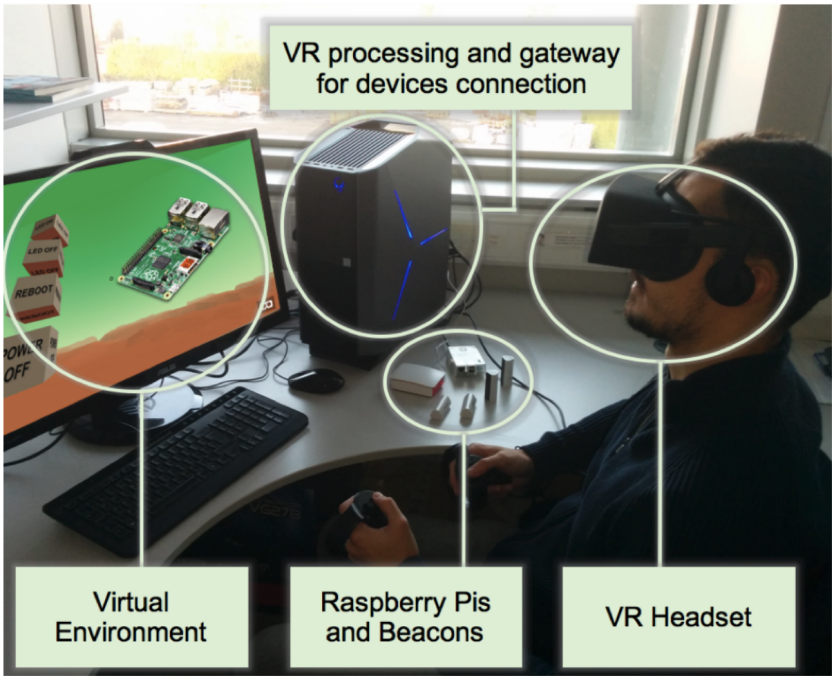


Figure 7.3: The implemented local network-based testbed

puter [259], which is connected wirelessly with the Raspberry Pis in a local 802.11n network, supports the Oculus Rift using an HDMI port, and also renders the browser-based 3D virtual world. The user shown in Figure 7.3 uses the Oculus Rift Touch controllers to select the available actions on the Oculus headset. This is replicated on the computer screen, which displays the same virtual environment seen on the headset. The actions (e.g. power off device, turn LED on, etc.) update the representation of the virtual Raspberry Pi displayed on the Oculus headset and are also executed on the real Raspberry Pis. The VR application also displays information (e.g. temperature) generated by the IoT beacons sent via Bluetooth.

The specifications of the Alienware computer, Beeks Beacons and the Oculus Rift used are available in Tables 7.1, 7.2 and 7.3, respectively.

The applications developed are as follows. The first application is a Java VR communication application, which was deployed on the Raspberry Pi, in order to read from and send commands to the in-board LED, read temperatures of the beacons connected to the Raspberry Pi via Bluetooth, and send the shutdown and restart instructions to the operating system of the Raspberry Pi. This Java application also communicates with the main computer (i.e. Alienware) using Java ServerSockets and Input/OutputStreams for receiving and sending instructions to the virtual IoT devices visualised in the VR headset.

Table 7.1: Alienware Specifications

Parameter	Value
Model	Alienware Aurora R6
Processor	Intel Core i7-7700
RAM Memory, Hard Drive and SSD	16GB, 1TB and 256GB
Graphics Card	NVIDIA GeForce GTX 1080 8GB
Operating System	Windows 10
3D Development	Unity Personal 2017.3

Table 7.2: Beeks Beacons

Parameter	Value
Battery	3.6V / 2600mAh - Primary Lithium
Size	2.36" x 0.85" (60mm x 21mm)
Weight	1.0 oz (28 gr)
Temperature Range	-30 to +77
Bluetooth Type	Bluetooth Low Energy 4.1
Bt. Sensitivity and Max. Power Output	-97dBm and +5dBm
Bluetooth Antenna	0dBm Single Antenna, Omni Directional
Bluetooth Data Rate	1Mbit/s / 2Mbit/s
Bluetooth Security	128 bit AES
Power Consumption RX	7.5mA RX Active Mode
Power Consumption TX	6.5mA TX Active Mode
Power Consumption Sleep	1.6μA
Power Output	-40dBm to +5dBm
CPU	Dual Code: ARM Cortex M3 and M0
Sensors	*High Accuracy Temperature sensor* *3 Axis Accelerometer* - Detect.: Freefall. Motion, Pulse, Transient - Custom detection: Door opening/closing with counter; human walking detection; driving detection, motor vibration learning *Magnetometer* - Custom detectable modes: Door opening and closing, Metal nearby trigger, car detection, electric motor, efficiency/torque *Light Sensor*
Internal Flash Memory	55KB Flash standard
LED	Red LED

Table 7.3: Oculus Rift

Parameter	Value
Display	PenTile OLED
Graphics	2160x1200 (1080x1200 per eye) @ 90 Hz
Sound	Integrated 3D audio headphones (user removable/exchangeable)
Input	6DOF (3-axis rotational tracking + 3-axis positional tracking) through USB-connected IR LED sensor, which tracks via the 'constellation' method
Controller input	Oculus Touch motion tracked controllers
Connectivity	HDMI 1.3, USB 3.0, USB 2.0
Weight	470g (1.04lb)

The Alienware computer is responsible for running a Glassfish 4.0 web server [260] with the second Java application, which reads and writes to a MySQL database containing the status structures presented in Section 7.2.2. These statuses are related to the real world IoT objects. This application updates the database with the received instructions from the Raspberry Pi Java application, and also sends instructions back to the Raspberry Pi.

A JavaServer Faces (JSF) web application [261], which also contains the VR application in HTML pages optimised for VR, is responsible for reading and writing the statuses in the database regarding the virtual objects. The JSF application receives and sends the current statuses of the virtual objects coming from the interactions of the user with the VR headset manipulating the options in the HTML pages optimised for VR.

Testbed development started with the creation of the two Java applications in the Glassfish web server (i.e. Alienware) and the Raspberry Pi, as seen in Figure 7.4. As mentioned earlier, threaded Java ServerSockets are used on both Glassfish server and Raspberry Pi, therefore several devices can connect to the Glassfish instance. The Java applications convert the actions performed by the users into objects, following the structure presented in the section 7.2.2, so they can be sent over to the other application (either on the Raspberry Pi or Alienware) and also stored in the database, by the use of the MySQL Java database driver. These objects are exchanged using the ServerSockets and Input/OutputStreams every 0.5 seconds. The objects carry all statuses for the five types of interactions that the web application enables, from vir-

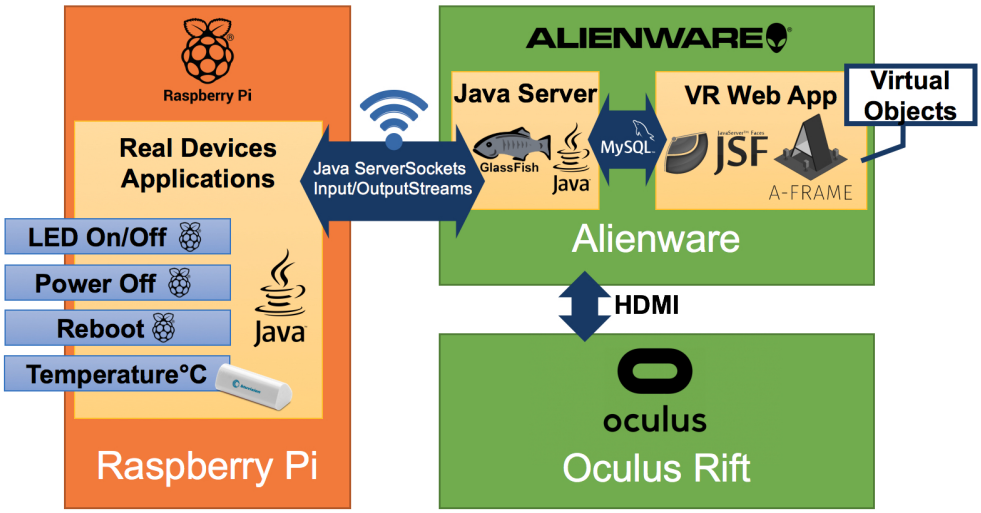


Figure 7.4: Local network-based solution architecture

tual devices (in the web application) to real devices and vice-versa. The five types of interactions are: turn LED on and off, reboot and restart, on the Raspberry Pi, and temperature measurement on the Raspberry Pi and beacons.

The 0.5s time interval for the local network testbed was selected based on the performance of the network and required communication time between real and virtual devices. Experimental tests did not show any significant improvement in the perception of the real-time execution of tasks for time intervals lower than this limit (i.e. after an action is taken in the virtual environment, it takes 0.5s for the action to be executed in the real environment, and vice-versa). However, any rate larger than 0.5s would impact severely the performance of the platform. Several update rates were tested in order to define a balance between user perception and performance (i.e. CPU and memory consumption), ranging from 0.1s to 2s. Update rates larger than 0.5s start to negatively impact the VR application. Such negative effects need to be avoided in order to prevent both poor quality and potential user motion sickness.

The developed JSF web application also runs on the Glassfish server and was built with responsive design for mobile device screens and VR devices. JSF is responsible for binding the user interface and the core of the Java application, which communicates with the IoT devices using sockets in the local network.

The HTML pages of the web application implement Mozilla's A-Frame [262], an HTML-based framework for VR development. A-Frame includes tags designed to enable head movements in smartphone-based VR headsets by natively reading smartphones' accelerometers and gyroscopes. It is also compatible with regular VR head-

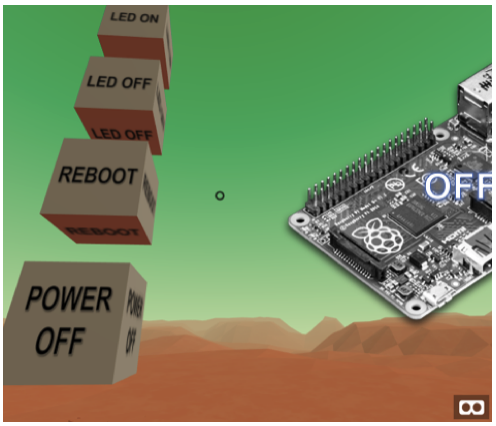


Figure 7.5: Virtual Raspberry Pi - Device off

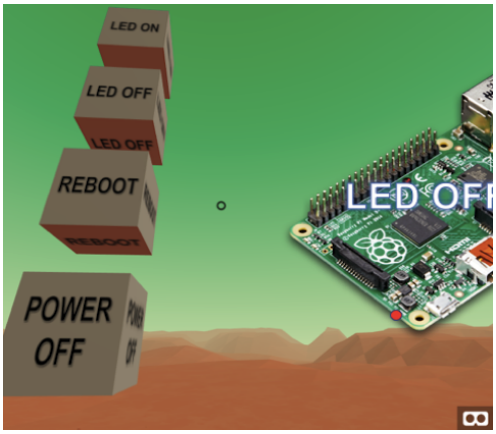


Figure 7.6: Virtual Raspberry Pi - LED off

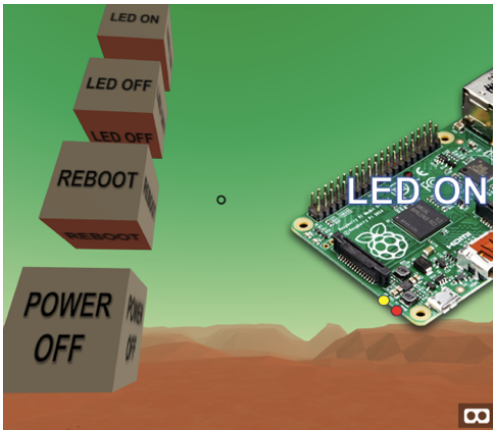


Figure 7.7: Virtual Raspberry Pi - LED on

sets. A-Frame supports and offers creation tools for 3D environments. It also duplicates graphical content for the use of smartphones inside of VR headsets such as the

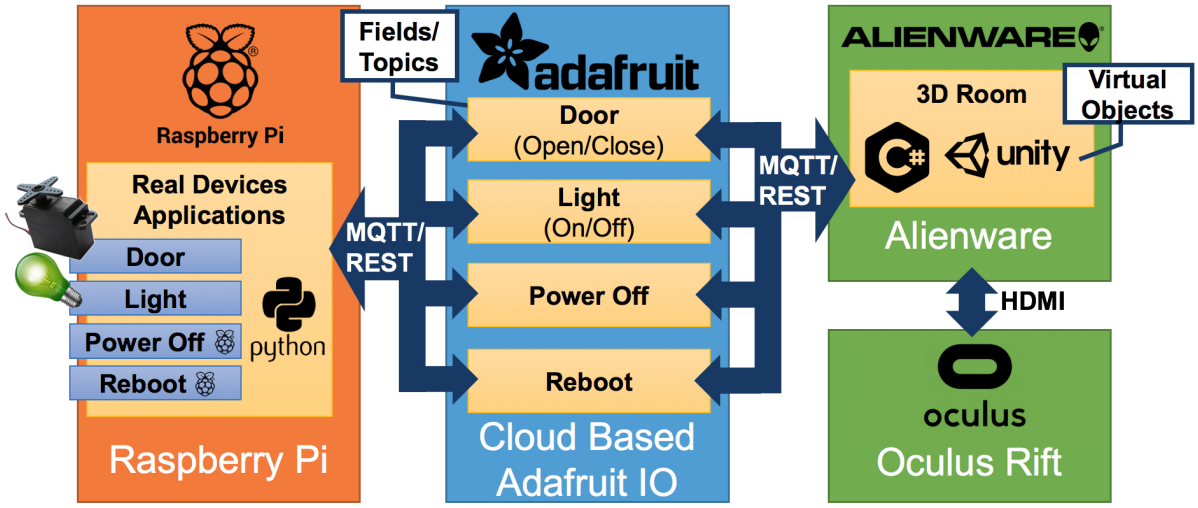


Figure 7.8: Cloud-based solution architecture

Google Cardboard (each half of the smartphone screen is seen by a different eye, creating a 3D effect with the help of the VR headset). A-Frame is compatible with web browsers such as Google Chrome and Mozilla Firefox, and also with the Oculus Rift.

Figure 7.5 illustrates how the Web-VR application displays the virtual device turned off. In Figures 7.6 and 7.7, the virtual device has its LED off and on respectively. Users are allowed to control objects which they are granted permission. This allows users to keep devices private or share them with different users, developing a social network of virtual and real IoT devices. In order to enable device sharing, users are identified by user IDs, allowing them to share their devices with other users, therefore, the web application has login and share functions.

The MySQL database stores all data necessary for the monitoring and synchronisation of the solution, with timestamps, IDs for users, devices and tasks, and the permission lists with users and their granted devices.

7.4.2 Cloud-Based Approach

The cloud-based implementation allows for the real-world devices to be visualised and manipulated through the use of the Oculus Rift VR headset. Remote IoT objects can be accessed and receive the VR actions performed, as the actions are stored and synchronised on the Adafruit IO cloud server, as detailed in Figure 7.8.

In this testbed, the major components are the VR headset (Oculus Rift) connected to an Alienware computer, a Raspberry Pi 3 Model B+, IoT objects (servo-motor –

Figure 7.9: Virtual living room - light on

Figure 7.10: Virtual living room - open/close door

representing a motor to open and close doors – and light bulbs), and the cloud-based Adafruit IO server. For the cloud-based approach, a 3D virtual room was designed using Unity, containing wall switches, which are used to open/close doors and to turn on/off lights in the real-world, just as they are performed in the virtual environment, as seen in Figures 7.9 and 7.10. Functionalities programmed using C# scripts, such as shutting down and rebooting the real-world system (i.e. Raspberry Pi), are also available in the VR application. Users interact with the 3D VR environment by using the Oculus Touch controllers.

The real-world part of the proposed solution contains a Raspberry Pi to which IoT objects, such as a servo-motor and a light bulb, are connected. The interfacing of

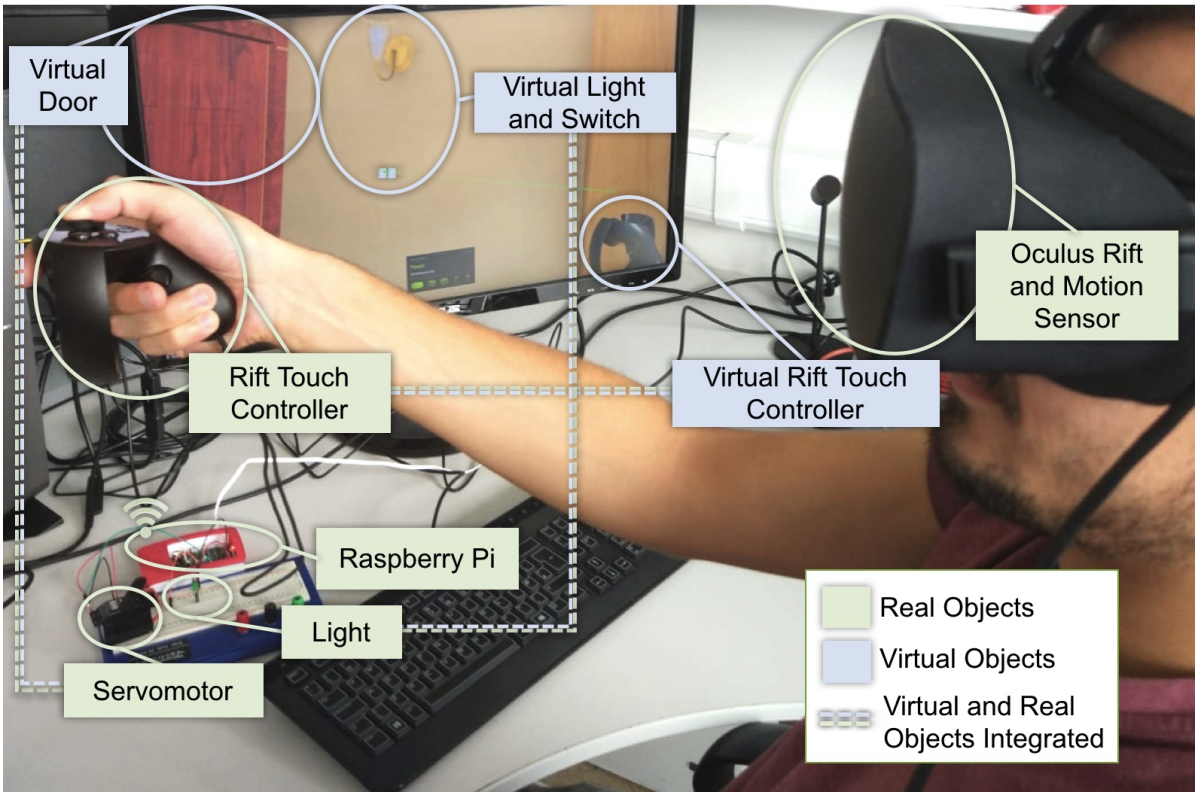


Figure 7.11: The implemented cloud-based testbed

the IoT objects to the Raspberry Pi is performed via the Raspberry Pi GPIOs and a program written using the Python programming language.

The cloud-based IoT server Adafruit IO is responsible for storing the commands received from the VR-world devices as well as the real-world devices. Adafruit IO manages "feeds" that are available in its dashboard. These feeds store the commands for each device. These commands are accessed by the devices via communications protocols (i.e. MQTT and REST) at lower network layers with instructions such as open/close for the door and turn on/off for the lights, or shutdown and reboot instructions. The Raspberry Pi is also connected to Adafruit IO, and therefore it reads and writes to the feeds related to the devices attached to the Raspberry Pi (e.g. light bulb, servo-motor), updating the feeds with the latest status values synchronised by VRITESS.

Whenever an action is performed by users in the virtual or real worlds, such as pressing switches for turning on/off lights or opening/closing a door, the commands DOOR_OPEN, DOOR_CLOSE, LIGHT_ON or LIGHT_OFF are sent to the specific feeds available in the Adafruit IO cloud server, according to the actions performed. A

menu is presented when pointing at devices such as doors or light bulbs, with the option of sharing these devices with other users. Users must be granted permissions for device manipulation and for sharing devices. Figure 7.11 presents the implemented cloud-based testbed in the Performance Engineering Laboratory.

7.4.3 Test Scenario Description

Tests conducted compared the local and cloud approaches and within the cloud approach, two protocols were tested and compared.

7.4.3.1 Local Approach vs. Cloud Approach

In order to keep the devices synchronised with their corresponding virtual or real devices, an implementation of communication protocols was necessary. Adafruit IO supports communications using MQTT or RESTful API. Adafruit IO has a current rate limit of 1 request per second (or 60 requests within 60 seconds), therefore lower latency is experienced in the local network approach, with a higher rate of 0.5s for sending/receiving messages by the use of Java ServerSockets.

Extra testing was performed to test the latency in the cloud-based approach. Using the tests performed in the local approach as a control test case, it could be observed that simple database queries ran much faster in the local approach. Twenty select queries were performed in both local and cloud-based implementations, and the average retrieval time was 4ms for the local test running MySQL and 12ms in the cloud-based approach powered by Adafruit IO.

7.4.3.2 MQTT vs. REST in Cloud Approach

Tests were conducted in both MQTT and RESTful API in order to demonstrate which approach has the best performance in the cloud-based implementation of the VR-IoT environment, in terms of communication delay and data traffic.

Table 7.4 shows the minimum, maximum, standard deviation and average delays of the ten times the application was executed, as seen in Figure 7.12 (sorted from minimum to maximum delay). Delay here refers to the amount of time required to transfer the action performed either in a real-world device or the virtual reality device to the cloud-based IoT server. The delay calculations were performed using Wireshark. The cloud-based IoT server, Adafruit IO, runs on the ports 1883 and 8883 (for SSL encrypted connections), and the 8883 port was used for the tests. In order

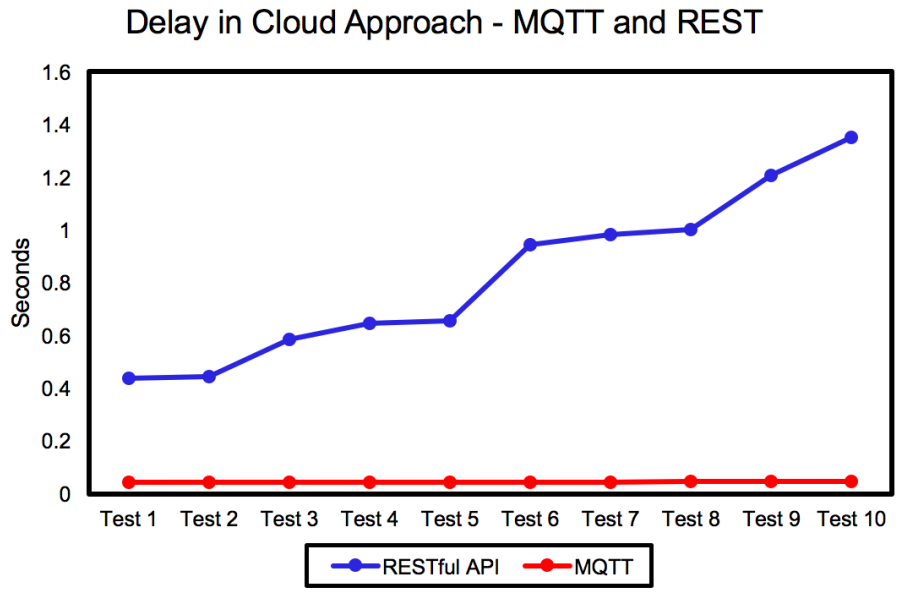


Figure 7.12: Delay (s) in cloud approach - MQTT and REST

Table 7.4: Delay (seconds) - MQTT and RESTful API

Delay (seconds)	RESTful API	MQTT Prococol
Maximum	1.353s	0.0469s
Minimum	0.439s	0.0419s
Average	0.826s	0.0437s
St. Dev.	0.301s	0.0017s

to monitor delay, Wireshark's TCP port was set to the cloud-based IoT server port, allowing the capture of all the outgoing packets, data, their time and length.

One of the major reasons that impact RESTful API's delay in comparison to MQTT is the fact that a new connection is established each time the devices send data to the cloud and then the connection is terminated. MQTT optimises this process, keeping the connection alive once it is established. Tests demonstrated that, in average, there is 20 times less delay in MQTT communications in comparison to the RESTful API in the cloud-based VR-IoT environment.

Table 7.5 shows the number of outgoing and incoming bytes used for data communications when performing actions exchange in the VR-IoT environment (i.e. bytes measured for the six available actions: turn on/off LEDs, turn on/off servo-motor, reboot and shutdown), also demonstrated in Figure 7.13 (sorted from minimum to maximum values). The MQTT and RESTful API implementations were compared in terms of outgoing and incoming data, respectively, in relation to the Raspberry Pi and

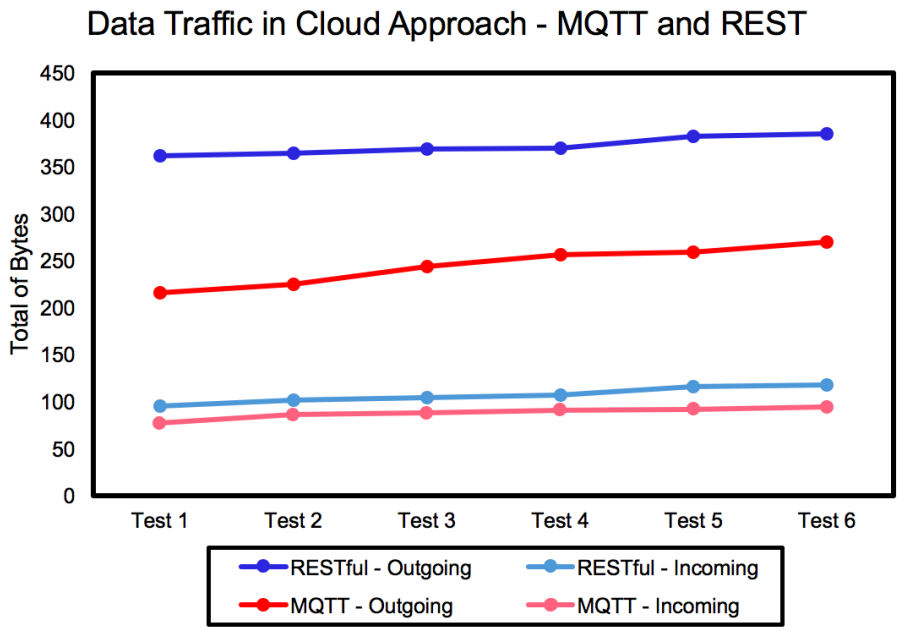


Figure 7.13: Data traffic (bytes) in cloud approach - MQTT and REST

Table 7.5: Data Traffic (bytes) - MQTT and RESTful API

Direction		RESTful API	MQTT Prococol	Avg. MQTT Improvement
Outgoing Bytes	Maximum Minimum Average St. Dev.	385B 362B 372B 8B	270B 216B 245B 17.7B	34% (in relation to REST outgoing traffic)
Incoming Bytes	Maximum Minimum Average St. Dev.	118B 95B 107B 7.4B	94B 77B 88B 5.2B	18% (in relation to REST incoming traffic)

Alienware communicating to the Adafruit server, and vice-versa. When considering average outgoing data, 245 bytes are used for MQTT in comparison with 372 bytes for REST and measuring average incoming data, 88 bytes are needed by MQTT versus 107 bytes by REST. Therefore, in terms of percentages about 34% less outgoing traffic and approximately 18% less incoming data is exchanged by the MQTT solution in comparison with the REST approach. This is also shown in Table 7.5. The main explanation for these results is the lightweight design of MQTT, which maintains one TCP connection alive and uses small size headers in comparison to the RESTful API that runs over HTTP, which creates new TCP connections when needed.

The positive results obtained by the use of VRITIP and VRITESS could be extended to include another interface for improving user experience, besides VR. Social networks provide APIs and features useful for remotely sending messages, and in the context of VRITIP, controlling IoT devices. The implementation of a testbed showcasing a social IoT network using Twitter is described in section 7.5.

7.5 Devices Virtualisation for a Social IoT Platform

A new paradigm known as the Social Internet of Things (SIoT) has emerged and it incorporates social networking concepts into IoT. These concepts include a structure with guaranteed route navigation, which will ensure scalability and effective discovery of connected objects and trust between connected objects and users that are 'friends' within the SIoT platform [108].

Based on these concepts, the devices virtualisation presented in the previous sections can provide a framework for additional interactions with devices, besides VR. A social IoT network was developed to demonstrate how user-friendly applications, such as Twitter, can be used to operate IoT devices remotely. A detailed presentation of the principle and testbed (illustrated in Figure 7.14), is included, as well as the associated algorithms, protocols, and cloud integration.

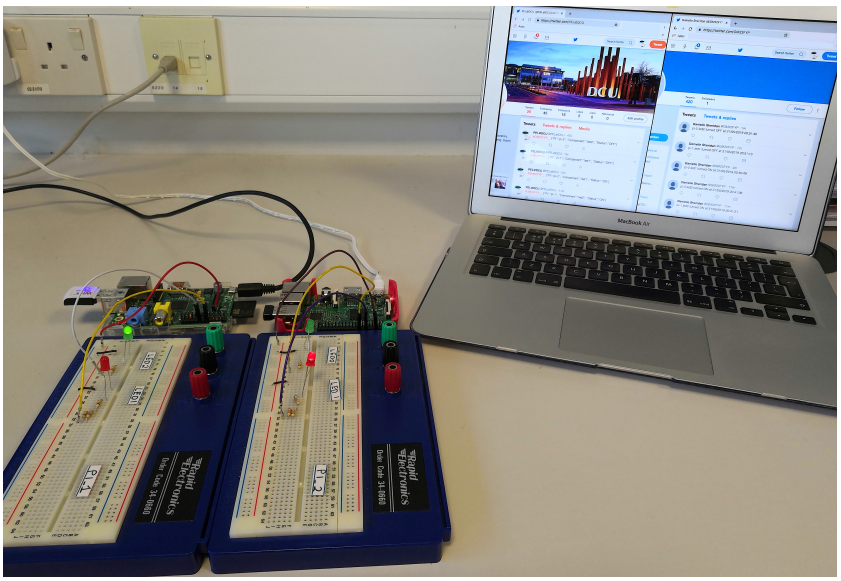


Figure 7.14: Twitter-based social IoT network testbed

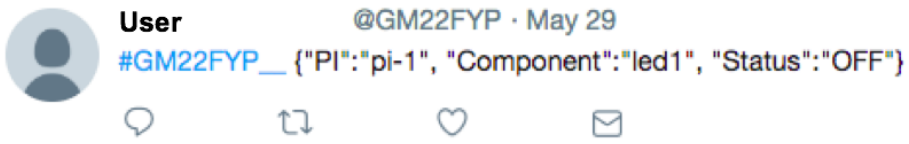


Figure 7.15: Tweet format

Figure 7.16: Tweet response from device

7.5.1 Social IoT Communication Structure

Once IoT devices are virtualised in the cloud through VRITIP, meaning that they can be identified and accessed in a virtual platform, different interfaces can be used to operate these devices, such as the previously described VR synchronisation, and also social networks.

The solution design of a Twitter-based social IoT network consists of mechanisms that allow users to tweet actions to be performed by IoT devices and receive tweets with notifications of status changes.

Users can interact with devices by writing tweets from any twitter account, as long as not blacklisted in the IoT gateway. Tweets need to follow a certain structure that can be translated into device actions containing relevant information such as, the device in which the user wants to access a component, the target component on that device and the action desired (e.g. turn on/off).

The use of the hash sign (#) at the start of a particular word or phrase is an essential part of 'Tweeting' known as 'hashtagging'. Hashtags are used to identify tweets that belong to a certain subject or category, and for this solution, it allows identification of messages sent to the smart IoT gateway of the platform.

Tweets with the correct hashtag and user action are converted into a JSON map, and device, component and status are the keys of the JSON map. The value for each key would relate to the chosen device, component or status update. For example, a user that wishes to turn on an LED on a device would tweet the following: #GM22FYP__ {"PI": "pi_1", "Component": "LED_1", "Status": "ON"}. This tweet information is then converted into a JSON map. An example of this structure being

tweeted can be seen in Figure 7.15, with a device response presented in Figure 7.16.

Twitter processes around eight thousand tweets sent every second. In order to obtain the desired tweets, they must be constantly searched, something possible thanks to the specified hashtag previously mentioned. Once the tweet hashtag matches the criteria the tweet data can be processed in order to complete the user operation. Tweet content is analysed so the expected map keys can be extracted. Once this is performed, the data is transmitted by a communications protocol to a cloud-based server. If a map or the appropriate keys are not found, the data will not be sent. Twitter uses the REST protocol for communications.

7.5.2 Social IoT Testbed

Similarly to the cloud-based implementation of the VR-IoT solution, the social IoT network uses Adafruit IO, which is an online cloud platform designed to store IoT data, as described earlier in this chapter. Adafruit supports data protocols such as REST and MQTT, and contains feeds representing the values (e.g. on and off) of components, such as LEDs on a Raspberry Pi. The data can be accessed and updated by a device by supplying the feed name.

Testbed also includes Raspberry Pis, several LEDs and breadboards. Two communications protocols were implemented in the solution for network analysis: MQTT and REST.

The solution design is illustrated in Figure 7.17. Design and implementation details are presented in this section. The implementation and the testbed allow for testing of the proposed architecture and solution performance analysis in a real scenario.

One of the Raspberry Pis acts as a gateway to bridge Twitter to Adafruit, running specific scripts for Twitter and Adafruit connection. The remaining Raspberry Pis act as IoT devices with components (e.g. LEDs) that can be controlled through Twitter, running scripts to read Adafruit's feeds and act accordingly. The gateway Raspberry Pi can also act as an IoT device and have components attached to it.

The gateway Raspberry Pi connects to Twitter, search for relevant tweets, analyse them, connects to Adafruit and send the appropriate data. The gateway also continuously checks for changes in the Adafruit feeds informing users about new status updates generated in the IoT devices.

Each Raspberry Pi acting as an IoT device contains two LEDs, red and green, which can be 'ON' and 'OFF'. These Raspberry Pis constantly check their corresponding

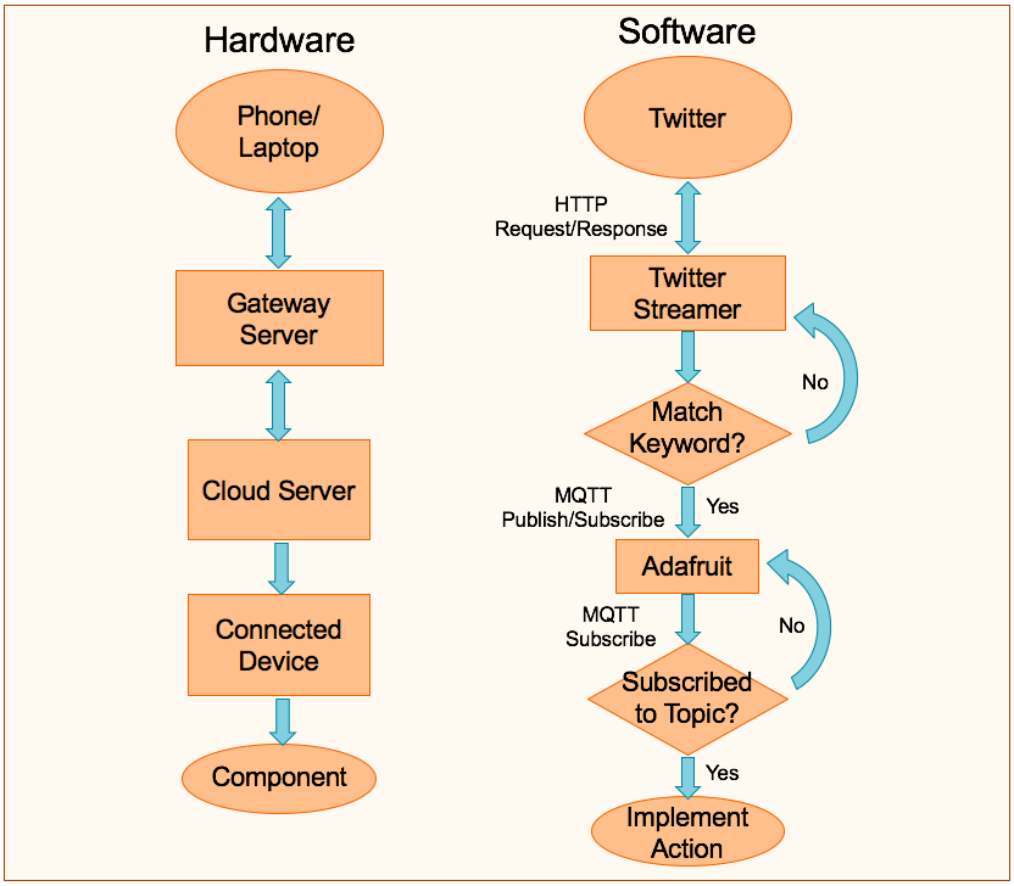


Figure 7.17: Solution design

feed on Adafruit and act appropriately responding to the latest user action.

In order to determine resistor values used for each LED, the voltage supplied is considered, which in the testbed is 5V. The voltage-drop across the LED is taken away from the supplied voltage, and for the red LED this value is 3.2V. The desired current that will flow through the LEDs must be about 25mA. Using Ohm's Law, the resistance value needed for the red LED is about 128 Ohms, which will be rounded to 150 Ohms. The green LED which has a voltage drop of about 3.3V so it needs a resistance value of around 68 Ohms, therefore, a 100 Ohm resistor will be used with the green LED.

Raspberry Pis' GPIO pins supply the power to the LEDs through resistors and a breadboard. The required pins are set as outputs. Values considered in the Raspberry Pis in order to update LEDs are 1 for on and 0 for off.

7.5.2.1 Twitter Streams Filtering

Twitter's API can be accessed with a developer account, in which the user is given OAuth authorisation to send requests to the API. Requests are sent over a TCP connection using a RESTful connection. The Twitter API contains several relevant endpoints that can be used for tasks such as tweet filtering in real time. The parameters for this endpoint are optional, and an example of its use is to track a specific keyword in a specific language. By selecting English in this parameter, it avoids misinterpretation of other languages. The programming language used in the scripts running in the gateway Raspberry Pi is Python, and a Python library called 'Twython' [263] is employed with the tasks of connecting to Twitter and setting up a stream (TwitterStreamer()). In order to avoid delays in searching for relevant tweets, the TwitterStreamer() is threaded and the tweets are put into a Queue() to be handled appropriately, as presented in Algorithm B.1, in Appendix B.

During the stream filtering phase, it is also possible to add security measures. The hashtag filtering is the first one, as only users who know the hashtag can control the objects. Additionally, filtering by username is also possible, as Twython receives the username related to all tweets. The gateway Raspberry Pi contains a list of 'whitelisted' users whose tweets must be processed, increasing network security. Twitter already performs a reliable authentication process during login.

7.5.2.2 Tweet Processing

Tweet content validation is performed before data is sent to Adafruit, so it receives data in the format it expects, as demonstrated in Algorithm B.2, in Appendix B.

The Twitter API sends responses in JSON format, encoding instructions in the 'text' field of the JSON response. An example of a JSON response when updating a status is presented in Figure 7.18. The keys of the JSON map are checked to ensure that all keys where inserted by the Twitter user (i.e. device, component and status). If the keys are present, then the data is sent to Adafruit.

7.5.2.3 Protocols Implementation

The gateway Raspberry Pi sends (i.e. maps of information from tweets into feeds) and receives (i.e. confirmation of insertion of data into feeds) data to and from Adafruit. Raspberry Pis used with IoT components read feeds on Adafruit. The communications of Raspberry Pis with Adafruit was implemented with the library Ada-

```
{
    "created_at": "Sun Apr 07 20:48:36 +0000 2019",
    "id": 1114993400862920705,
    "id_str": "1114993400862920705",
    "text": "'",
    "truncated": false,
    "entities": {...},
    "source": "<a href=\"https://GMMFYP.com\" rel=\"nofollow\">GMMFYP_1</a>",
    "in_reply_to_status_id": null,
    "in_reply_to_status_id_str": null,
    "in_reply_to_user_id": null,
    "in_reply_to_user_id_str": null,
    "in_reply_to_screen_name": null,
    "user": {...},
    "geo": null,
    "coordinates": null,
    "place": null,
    "contributors": null,
    "is_quote_status": false,
    "retweet_count": 0,
    "favorite_count": 0,
    "favorited": false,
    "retweeted": false,
    "lang": "und"
}
```


Figure 7.18: Twitter JSON response

fruit_IO_Python [264], and both REST and MQTT were employed for testing, as both protocols' implementations are available in the library.

The gateway Raspberry Pi sends the data received from Twitter to Adafruit, in the appropriate format. Once the client (i.e. gateway Raspberry Pi) is initialised and connected to Adafruit, there are functions available to be used according to the actions of the client, and shown in Algorithm 7.2.

For instance, when the client connects, the function "connected()" is responsible for subscribing the client to the appropriate feed. There are other functions for disconnections and updates from Adafruit when topics are changed and the new status is tweeted to inform the relevant audience. The send_message_to_adafruit() function sends data to Adafruit. The MQTT client package contains the function loop_background(), which creates a new thread that continuously listen for changes in feeds without disconnections.

The connected devices that are being controlled through Twitter, listen continuously for updates in the feed that they are subscribed to. For instance, in the testbed, the first Raspberry Pi will subscribe the feeds where "pi-1" is featured, as demonstrated in Figure 7.19, which shows the feed "pi-1" and two subgroups, "led1", "led2". If a value of 1 or 0 is presented in the feed "pi-1.led1", the LED will reflect this, 1

Algorithm 7.2 Cloud Communications

```
def connected():
        subscribe(feeds)
def disconnected():
        system.exit()
def message(feed, payload): //only for Gateway
        tweet_change_in_component(feed, payload)
def message(feed, payload): //only for Connected Device
        change_status(payload)
def send_message_to_adafruit(tweet): //only for Gateway
        if(Component==led1)
                if(status=="ON")
                        send_to_Adafruit(component, {value: 1}, feed)
                elif(status=="OFF")
                        send_toAdafruit(component, {value: 0}, feed)
        if(Component==led2)
        if(status=="ON")
                        send_to_Adafruit(component, {value: 1}, feed)
                elif(status=="OFF")
                        send_toAdafruit(component, {value: 0}, feed)
def main:
    createClient(Adafruit_username, Adafruit_key)
    if (connected_to_client)
                loop_background()
    else
                print(error)
```

represents on and 0 represents off.

7.5.2.4 Testbed Analysis and Results

For testing purposes, the time between sending of the tweets and the acknowledgment were recorded to determine the average, maximum and minimum time in which the user would perceive instructions being implemented.

Initial tests consisted of several tweets being sent at a low frequency to ensure they were processed with no packet loss. The timestamps were computed to determine maximum, minimum and mean time of transactions. The frequency of tweets being sent was then increased in order to determine latency and packet loss at different rates. Timestamps were computed just as soon as a tweet was received from Twit-

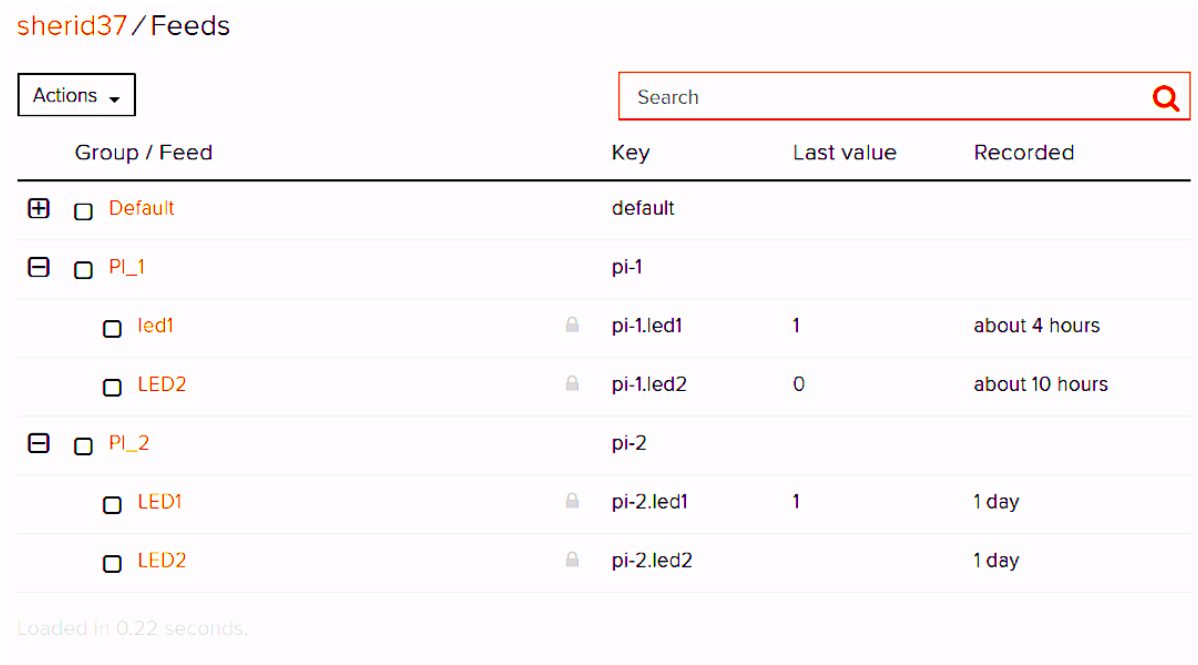


Figure 7.19: Adafruit feeds page

ter, before data being sent to Adafruit, and then another timestamp was recorded when data was received from Adafruit. Frequencies analysed were every 5 seconds, 2 seconds, 500 milliseconds and 100 milliseconds.

The graphs presented in Figures 7.20, 7.21 and 7.22 show average, minimum, and maximum latency, respectively, considering the time tweets were received by the Twitter streamer before publishing it to Adafruit, plus the time needed for the data to be sent to Adafruit and be updated in the feed. These tests considered REST for Twitter and MQTT for Adafruit connections.

Each tweet sent was processed by the scripts, demonstrating that there was no packet loss in these scenarios. A limitation of using Twitter's API for real time data only allows 1% of the tweets containing the search term being retrieved, which did not impact testing due to the hashtag used being very specific for the testbed and not having widespread use.

From the results from the graphs in Figures 7.20, 7.21 and 7.22, it is possible to notice that tweets are processed with varying latency. For instance, when a few hundred tweets were sent every 5 seconds, the minimum latency observed until the feed in Adafruit was updated was less than 0.3s whereas the maximum latency was around 0.6s. If the tweet frequency is very high, for instance, one tweet every 100ms, the latency varied from 0.33s (min.) to 34.6s (max.). The results show that the

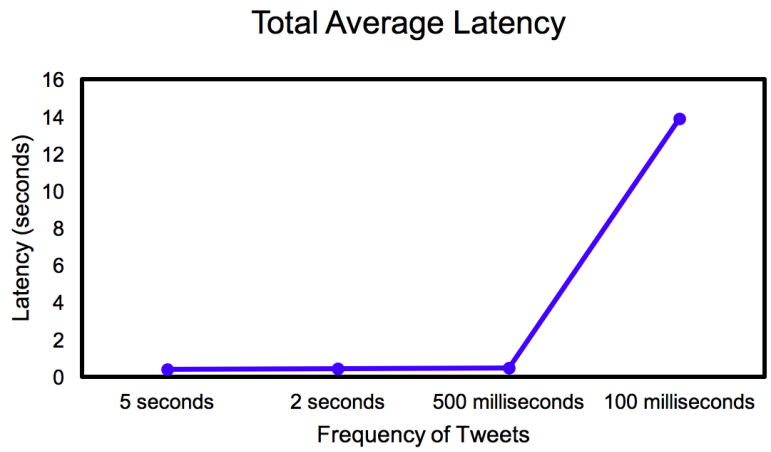


Figure 7.20: Average total latency

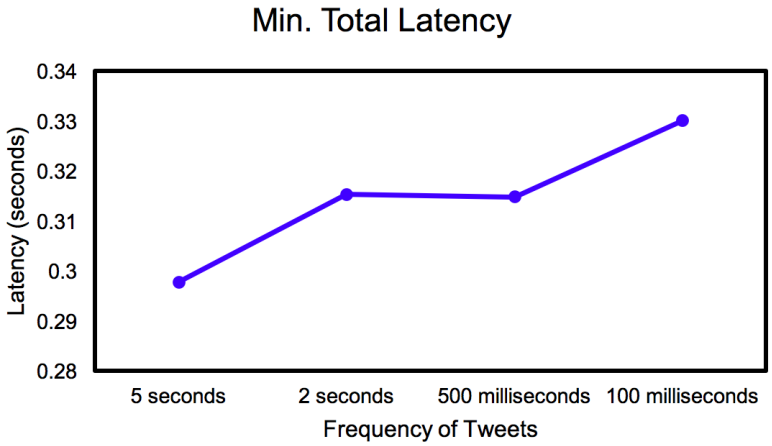


Figure 7.21: Minimum total latency

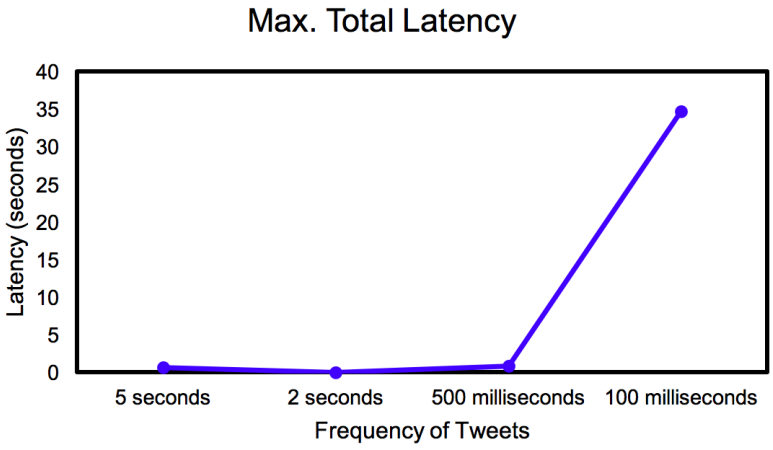


Figure 7.22: Maximum total latency

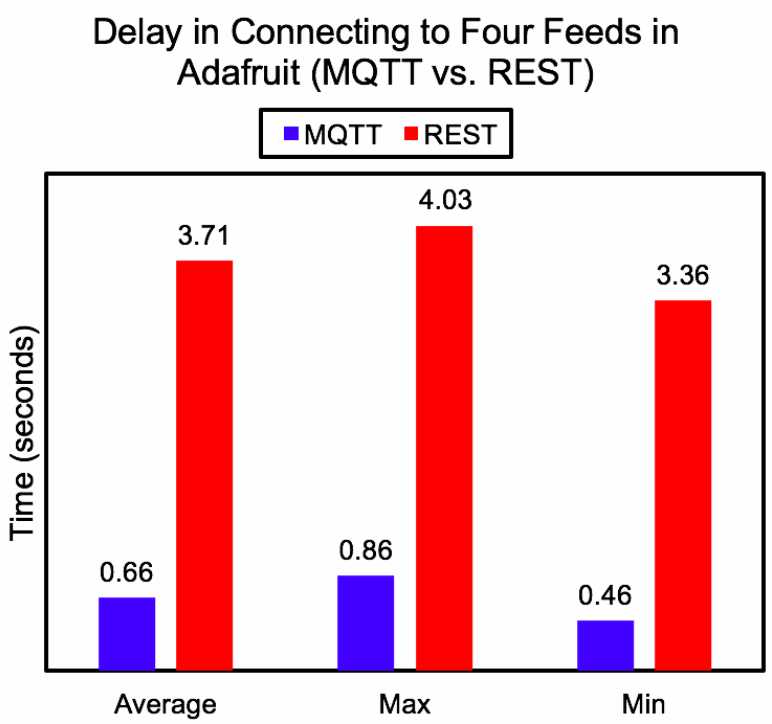


Figure 7.23: Delay in connecting to four feeds in Adafruit (MQTT vs. REST)

latency is low if tweets are sent every 500ms, 2s and 5s, but can be very high when tweets are sent every 100ms, due to high processing in the gateway Raspberry Pi and delays in the connections with Twitter and Adafruit. Therefore, a scenario with tweet frequency of 100ms is not recommended, as the maximum latency observed was 43 times higher than the maximum latency observed in the 500ms frequency (34.6s versus 804ms).

For protocol comparisons, the gateway Raspberry Pi communicated with Adafruit using REST and MQTT. Delay was measured for both protocols, as seen in Figure 7.23, for the connection with the four feeds available on Adafruit. MQTT performed much better than REST, as REST creates new connections every time it needs to transmit data. On average, in order to connect to the four feeds on Adafruit, the gateway Raspberry Pi experienced 82% less delay on communications when using MQTT instead of REST.

7.6 Chapter Summary

The innovative and novel VR-IoT Environment Synchronisation Scheme (VRITESS) was introduced and described. The proposed VR-IoT platform allows users to operate IoT objects in a virtual environment and contains a synchronisation algorithm that maintains virtual and real IoT objects updated, according to actions and events that happened in both virtual and real environments, reflecting changes on each other.

In order to demonstrate the benefits of the solution, two approaches were employed for the VRITESS real-life testing: a local network-based solution and a cloud-based deployment with a 3D room developed with Unity. Testing results show that the cloud-based solution has higher latency in comparison to the local-based approach. In the cloud-based scenario, two communications protocols were employed: MQTT and RESTful API. Testing results demonstrated better performance in favour of MQTT, as it has achieved lower delay and requires less amount of data exchanged due to its lightweight design.

Through device virtualisation, achieved by VRITIP, it was possible to develop further work on the design, implementation and analysis of a social IoT network, using Twitter for experiments. The social IoT network has the novelty of allowing users to operate IoT devices from a well-known interface and also providing the ability of device sharing among users. MQTT and REST protocols were examined and tests demonstrated that MQTT performs with much less delay when connecting to the cloud server. Raspberry Pis were controlled by tweets turning on and off LEDs and that allowed further analysis on the best scenarios for the solution, such as tweet frequency which perform well up to 500ms, demonstrating the benefits and usability of a social IoT solution.

This chapter concludes the contributions presented in this thesis, which aim to increase performance of IoT networks, as presented in the previous two chapters and improve IoT usability, as presented in this chapter. The next chapter recapitulates the contributions presented in this thesis and remaining challenges for future work.

Chapter 8

Conclusions and Future Work

This chapter summarises the contents of this thesis. A recapitulation of the contributions of the thesis will be presented, with the main results and outcomes outlined. Following that, directions and ideas for future works will be proposed.

8.1 Overview

The extremely large number of devices available to the modern day user, with the increase in device intercommunication, is fuelling the latest IoT developments. IoT needs to enable exchange of various types of data, from sensor data to multimedia, between numerous diverse devices differing in power, connectivity, mobility and energy, while also maintaining high levels of QoS. There are several challenges that IoT impose to researchers that go beyond maintaining good performance of devices and service delivery. Innovative ideas for user experience and device intercommunications are also important and must be explored.

8.2 Contributions

The works presented in this thesis aim to address challenges regarding performance and user experience in IoT networks, as mentioned above.

Firstly, a thorough survey of background technologies was necessary for an understanding of the current situation of the field. Several topics were covered such as sensor technologies, hardware and software for IoT, device identification, communications and protocols, IoT services and security, rich media IoT and the use of IoT with virtual reality, multimedia and social networks.

Secondly, a review of state-of-the-art solutions and applications was performed, and research works were categorised in IoT architectures; Performance-oriented schemes for IoT such as QoS approaches, age of information analysis, solutions for device location and relevance, device-to-device and clustering techniques, and cloud solutions for IoT; Solutions related to mobility; and Rich media IoT schemes, with additional use-cases being described.

Next, after the identification of research gaps, the research could be directed into three specific contributions, described below:

1. NETSMITS (NETworking Scheme for sMart IoT gatewayS) supports increased number of inter-communicating IoT objects and maintains higher quality of their offered services. NETSMITS employs assessing the values of QoS and Relevance metrics at the level of devices and gateways connected to a cloud-deployed IoT Integration Platform (ITINP). NETSMITS was assessed through NS-3 simulations and compared against another mechanism (i.e. Cluster-based Framework). Results demonstrate that NETSMITS is able to avoid packet loss, allow highest throughput per device to be achieved and reduce delay by 85%, by rearranging devices in the available gateways. It also enables with 63% more devices to be connected to the better performing gateway. A statistical performance study of NETSMITS was also performed. Findings show that delay is very similar in devices with different bit rate requirements, while throughput and packet loss affect mostly the low bit rate devices, giving an advantage to the high bit rate devices, which is reflected in terms of quality. This analysis is extremely important for the development of efficient IoT solutions, such as algorithms for routers and gateways, that are expected to interconnect a large number of devices. These solutions will need optimal configurations and adaptability in terms of QoS metrics, in order to provide the best quality for the highest number of devices possible. A statistical method, such as ANOVA, could be applied in real time to IoT algorithms in order to guarantee that the best clustering tasks based on QoS metrics are performed.

2. REMOS-IoT (RElay and MObility Scheme for the Internet of Things) is employed on a D2D-enabled architecture for IoT devices. REMOS-IoT contains algorithms and records scores for QoS, relevance, age of information and location, in order to cluster IoT devices efficiently and support mobility. The testing scenarios consist of a WiFi-only smart watch in a smart house with several other

smart devices initially connected to a local gateway using a WiFi 802.11ac access point. The smart watch loses connection and needs to relay on a mobile device before reconnection, which triggers REMOS-IoT algorithms. The solution was tested via NS-3 modelling and simulations and results for uplink and downlink were recorded in terms of throughput, packet loss and delay. The proposed solution also outperforms another baseline solution in terms of packet loss and delay.

3. VRITESS (VR-IoT Environment Synchronisation Scheme) and the proposed VR-IoT Platform (VRITIP) allow users to operate IoT objects in a virtual environment and contain a synchronisation algorithm that maintains virtual and real IoT objects updated, according to actions and events that happened in both virtual and real environments, reflecting changes on each other. Two approaches were employed for the VRITESS real-life testing: a local network-based solution and a cloud-based deployment with a 3D room developed with Unity. Testing results show that the cloud-based solution has higher latency in comparison to the local-based approach. In the cloud-based scenario, two communications protocols were employed: MQTT and RESTful API. Testing results demonstrated better performance in favour of MQTT, as it has achieved lower delay and requires less amount of data exchanged due to its lightweight design. VRITIP also allowed the design, implementation and analysis of a Twitter-based IoT Network. The virtualisation of devices enabled the use of APIs from the social network to be integrated into VRITIP, so devices could be controlled remotely. MQTT and REST protocols were examined and tests demonstrated that MQTT performs with much less delay when connecting to the cloud server. Raspberry Pis were controlled by tweets turning on and off LEDs and that allowed further analysis on the best scenarios for the solution, such as tweet frequency which perform well up to 500ms.

8.3 Future Work

1. Even though NETSMITS had a very positive testing outcome, additional innovations are possible for the solution. Future work for NETSMITS include prioritisation per device type. Metrics for energy consumption can also be included, bringing new challenges for overhead control and how often the algorithm must

be executed. A statistical-based approach could also be implemented during execution time for further QoS improvements through pattern analysis and prediction.

2. REMOS-IoT additional features could include an optimisation of the relevance score by prioritisation of different types of services and content. Energy-aware approaches could also be integrated and used during relay operations, for instance, devices with low power would not be selected for relay.

3. VRITESS and VRITIP could be further extended to accommodate additional types of smart devices, beacons and sensors into the virtual world. A testbed involving a massive number (e.g. hundreds or thousands) of IoT and VR devices interconnected in a social IoT network could also be envisaged. The social IoT network could also be expanded to seamlessly integrate other social networks, such as Facebook, Telegram and WhatsApp.

8.4 Final Remarks

According to the work presented in [265], IoT data can be used to create novel ways of connecting people, with automated software networking people and objects. IoT objects such as smart watches, wrist bands, appliances, healthcare devices and cars can bring new ways of user interaction. In [266], the author affirmed that VR will change fundamental rules on how life has been lived throughout the entire length of human history, creating social consequences.

Therefore, VR provides a rich media experience that increases the sense of reality with auditory, graphical and tactile capabilities that translate the real world into a virtual one. The possibility of manipulating virtual devices using intuitive controls and human senses such as tactile gestures and language can make it easier for users to operate and interact with real IoT objects.

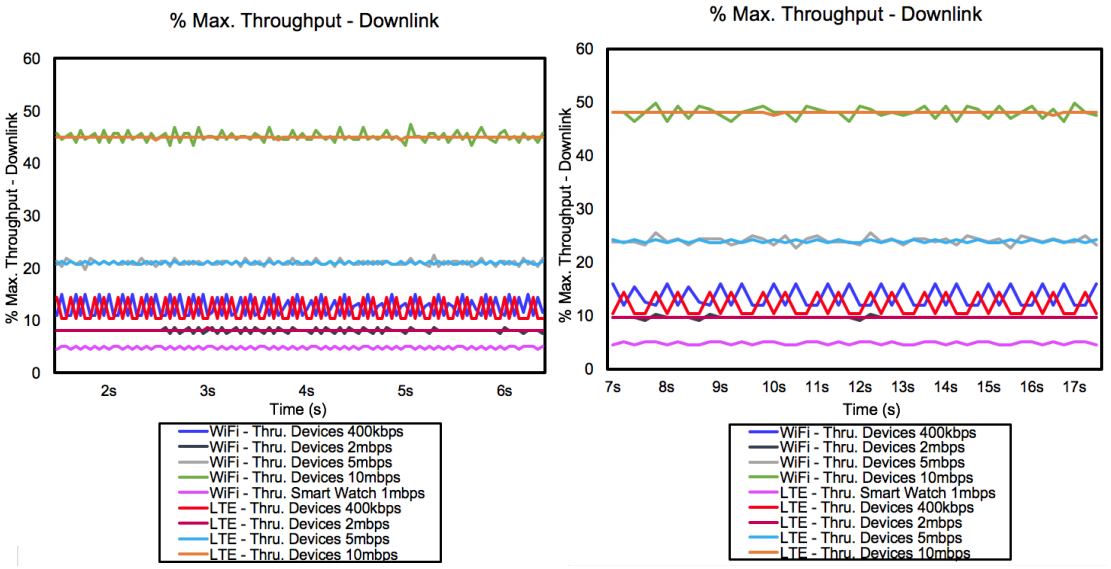
Additionally, the ability to share devices brings people together, and enables IoT services to be made accessible to users who do not possess certain devices. IoT networks with social features push even further the need for seamless and real-time experiences, so solutions such as smart clustering, D2D and relays, and synchronised device virtualisation are vital for IoT.

Bringing IoT and VR technologies together can expand their societal impact and help each other become more user-friendly and mainstream to society.

Appendix A

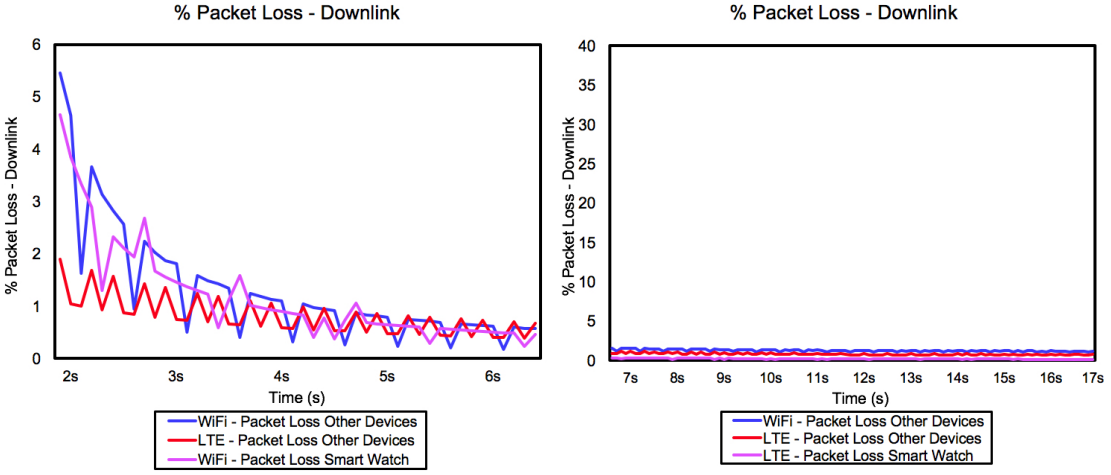
The figures in this appendix contain two plots to demonstrate network throughput, delay and packet loss, before (a) and after (b) mobile devices moved from the WiFi network to LTE, as described in the REMOS-IoT scenarios, in chapter 6, section 6.4.

A.1 REMOS-IoT Scenario 1



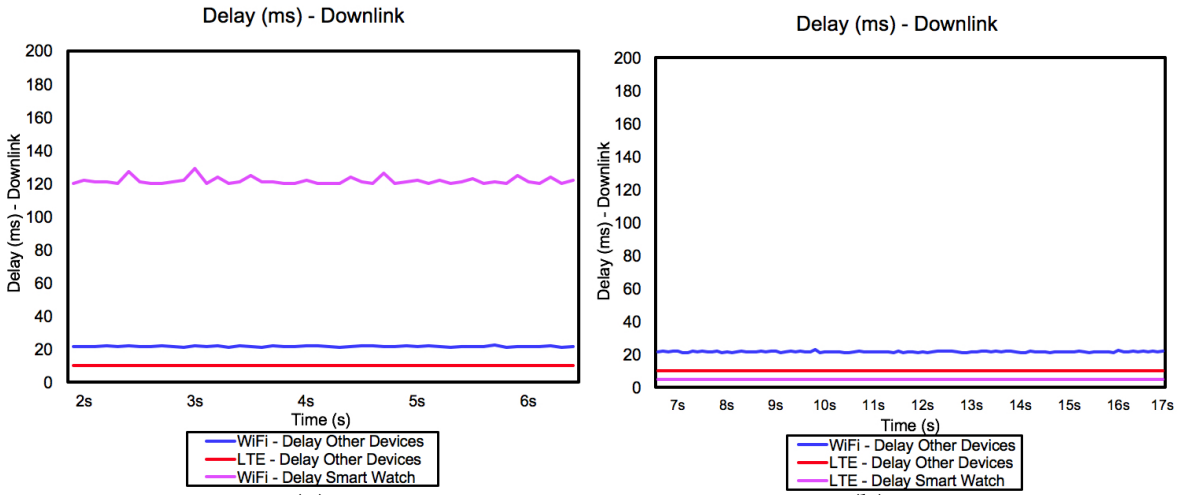
(a) (b)

Figure A.1: Low load 22 devices - Max. throughput results for downlink (%)



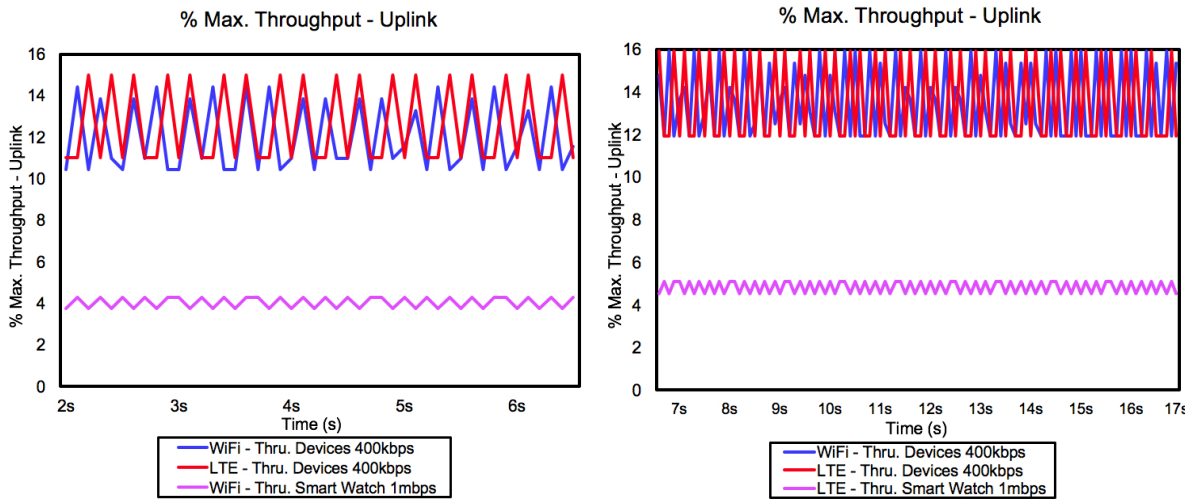
(a) (b)

Figure A.2: Low load 22 devices - Packet loss results for downlink (%)



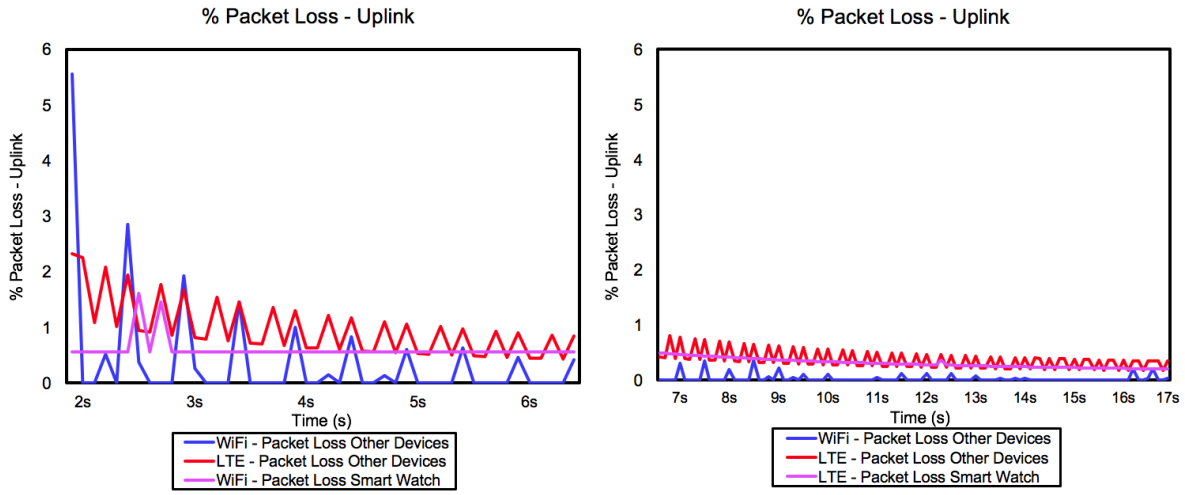
(a) (b)

Figure A.3: Low load 22 devices - Delay results for downlink (ms)



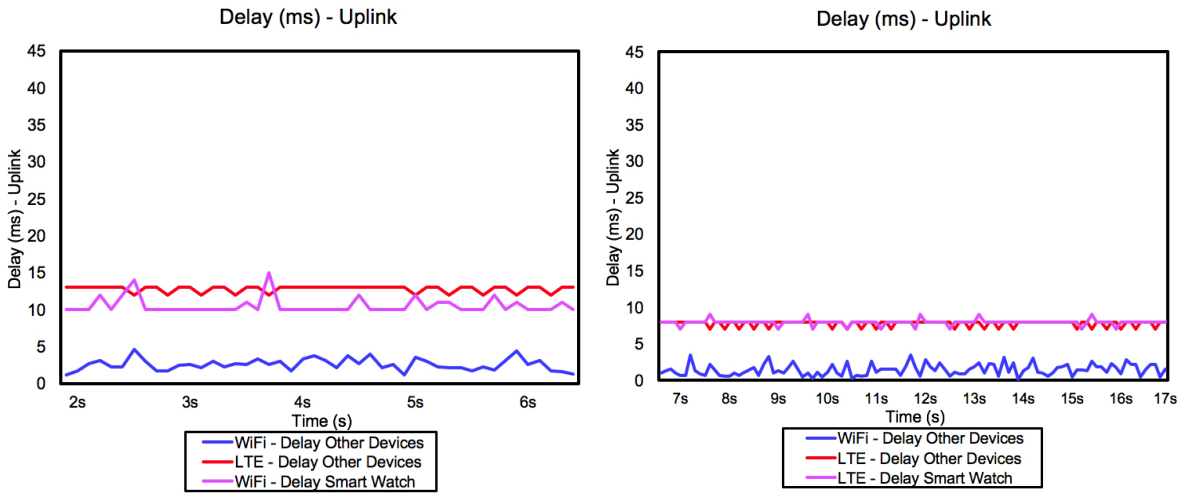
(a) (b)

Figure A.4: Low load 22 devices - Max. throughput results for uplink (%)



(a) (b)

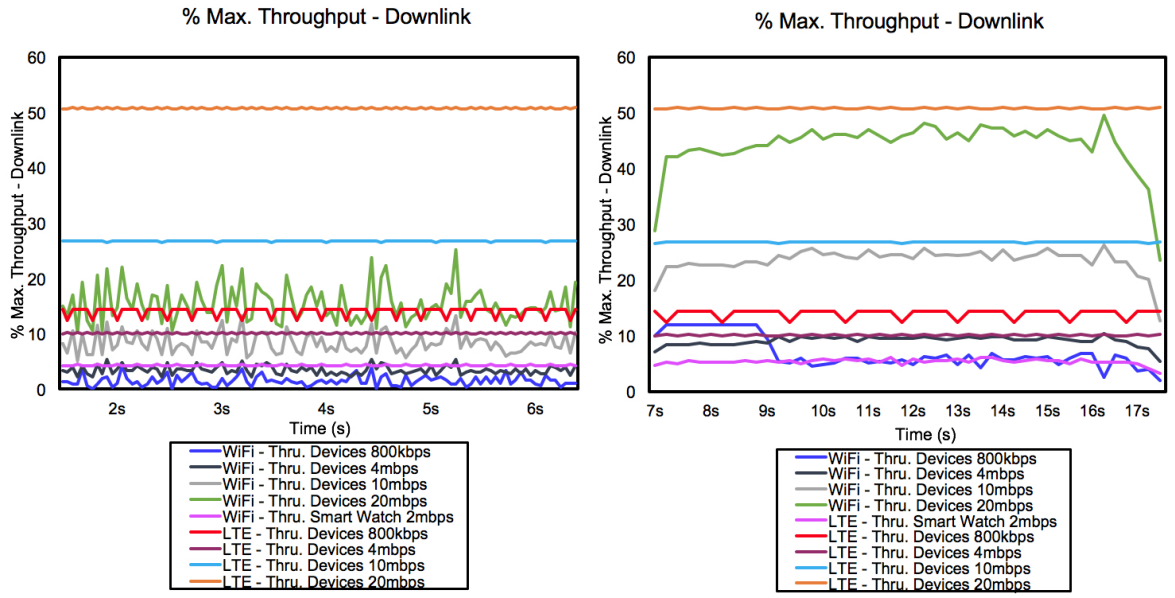
Figure A.5: Low load 22 devices - Packet loss results for uplink (%)



(a) (b)

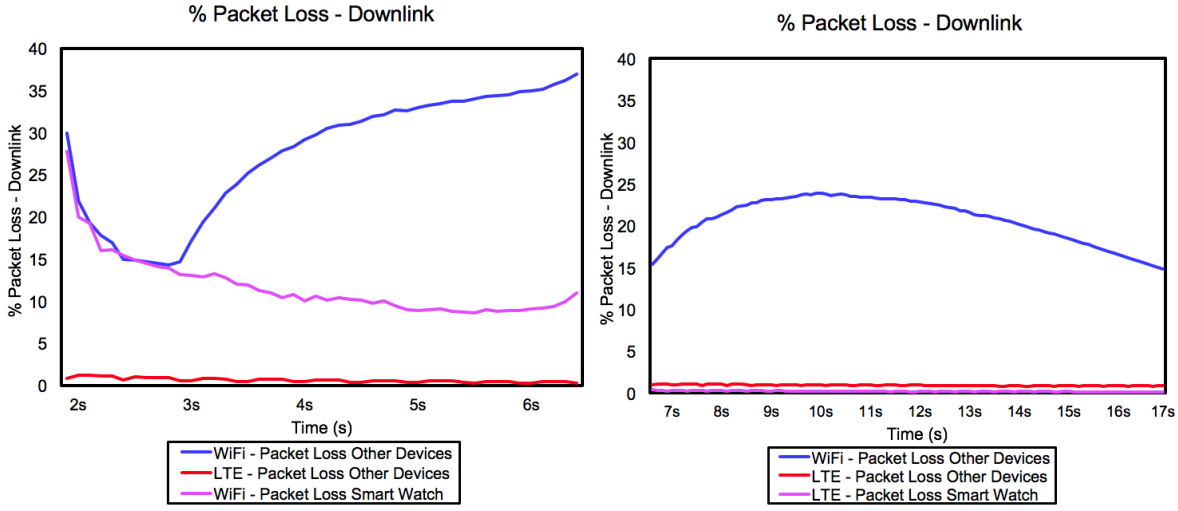
Figure A.6: Low load 22 devices - Delay results for uplink (ms)

A.2 REMOS-IoT Scenario 2



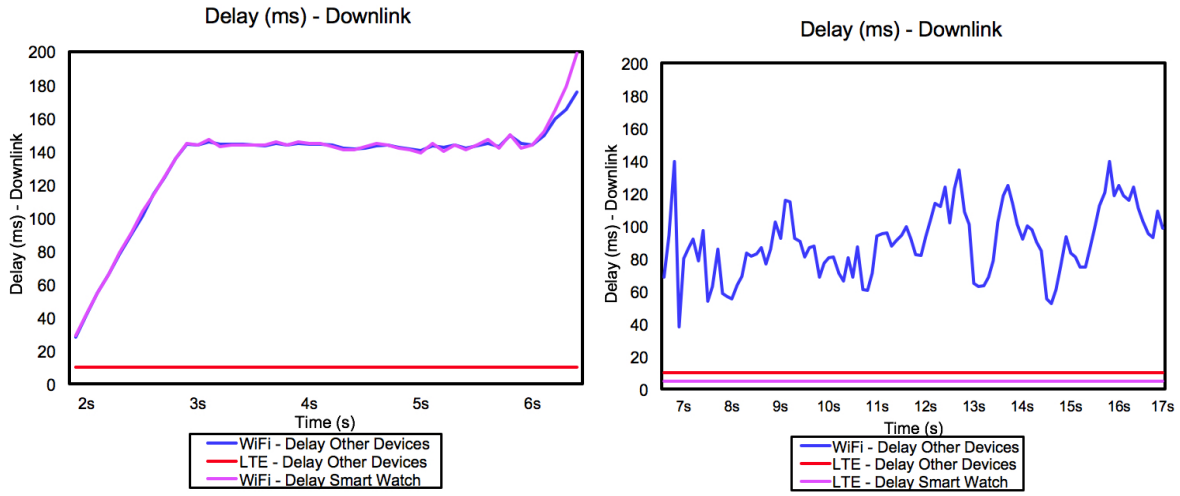
(a) (b)

Figure A.7: Average load 22 devices - Max. throughput results for downlink (%)



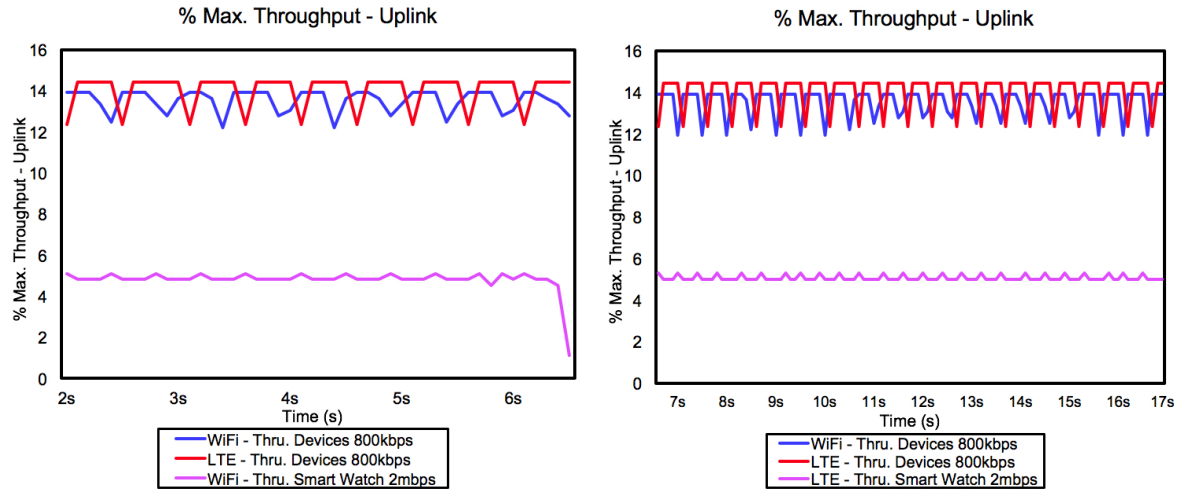
(a) (b)

Figure A.8: Average load 22 devices - Packet loss results for downlink (%)



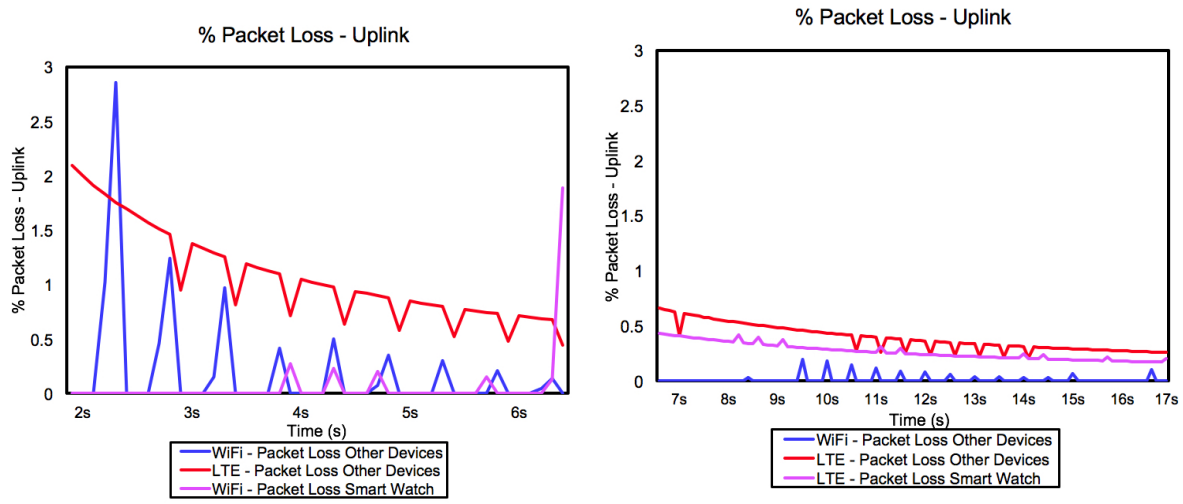
(a) (b)

Figure A.9: Average load 22 devices - Delay results for downlink (ms)



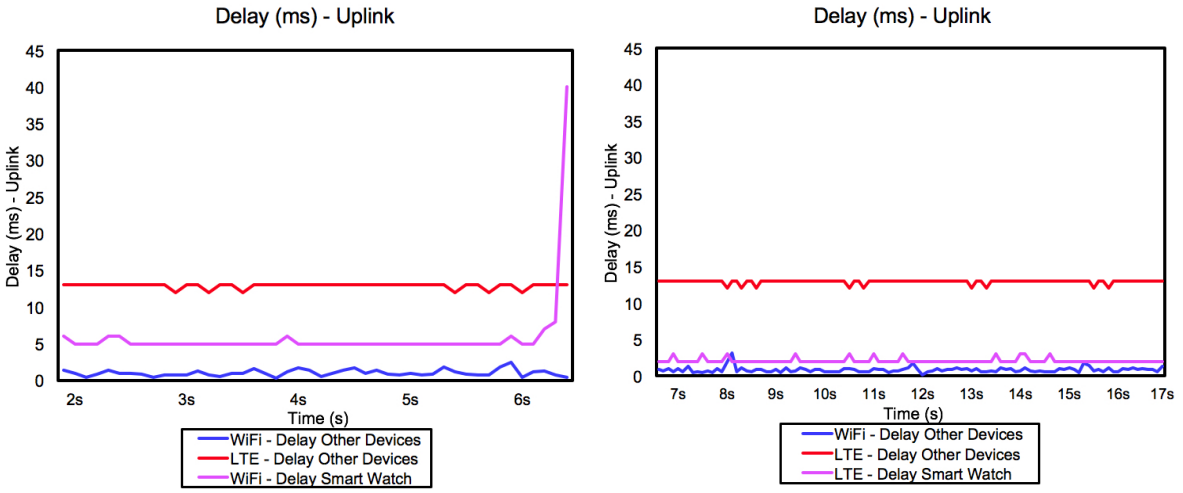
(a) (b)

Figure A.10: Average load 22 devices - Max. throughput results for uplink (%)



(a) (b)

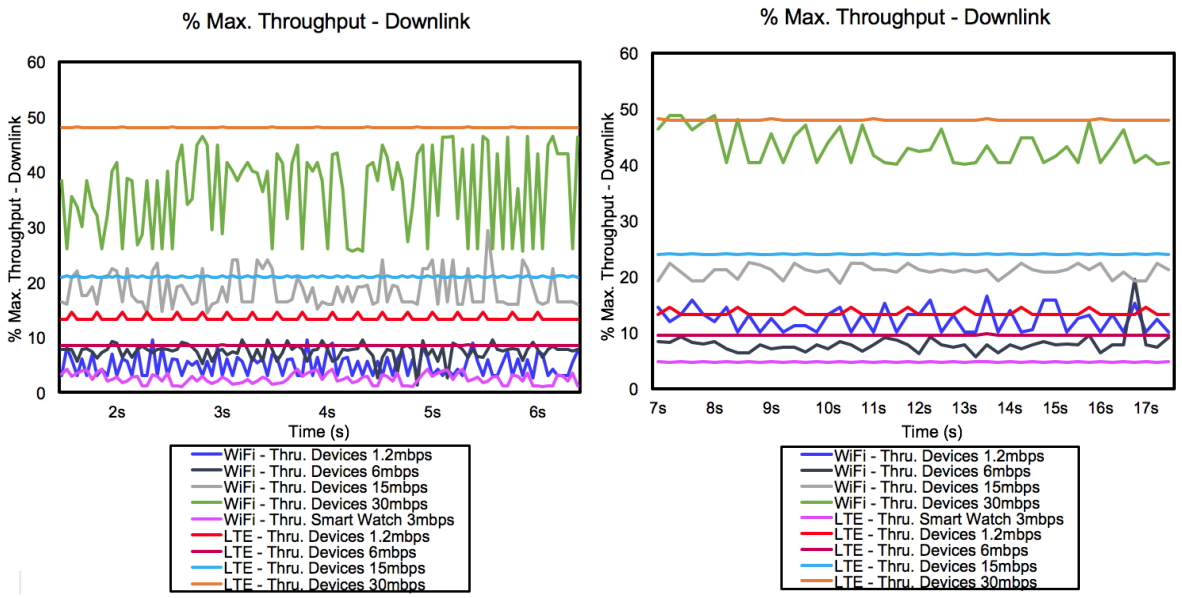
Figure A.11: Average load 22 devices - Packet loss results for uplink (%)



(a) (b)

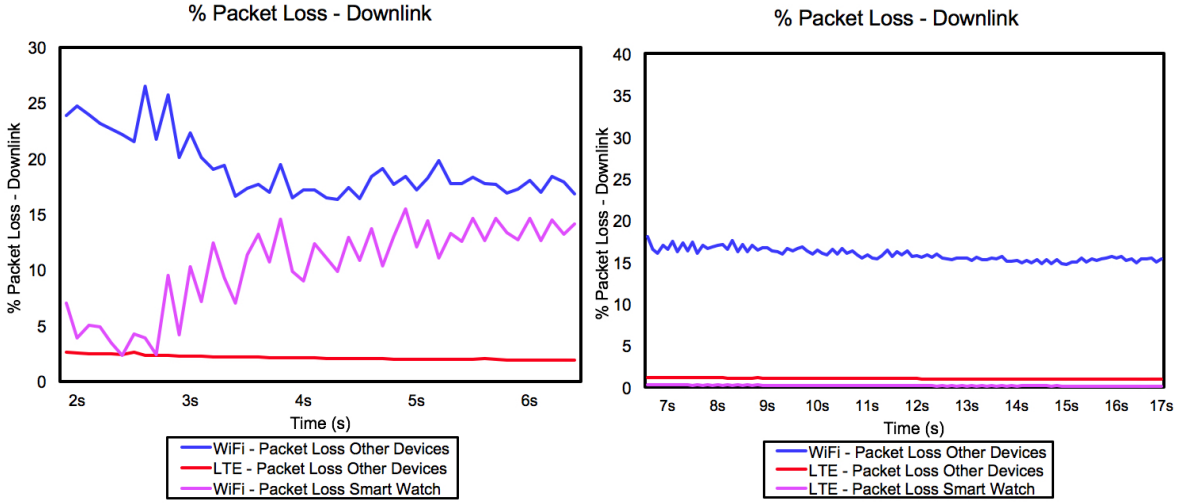
Figure A.12: Average load 22 devices - Delay results for uplink (ms)

A.3 REMOS-IoT Scenario 3



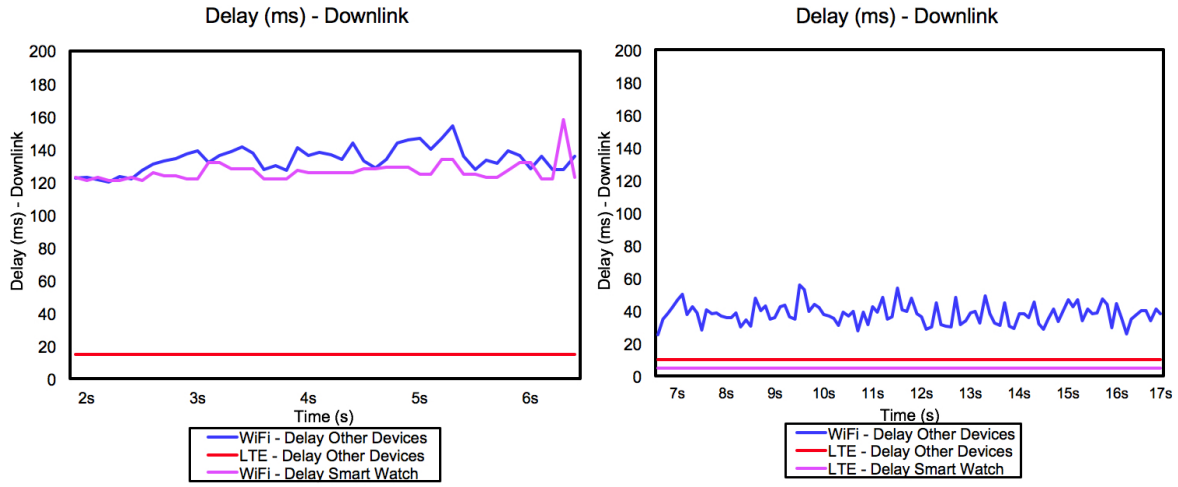
(a) (b)

Figure A.13: High load 22 devices - Max. throughput results for downlink (%)



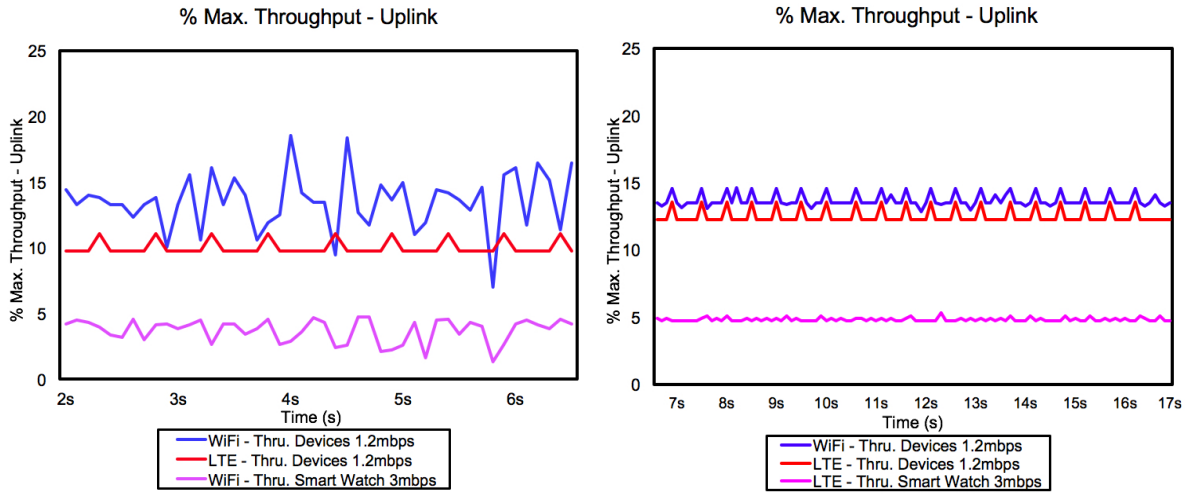
(a) (b)

Figure A.14: High load 22 devices - Packet loss results for downlink (%)



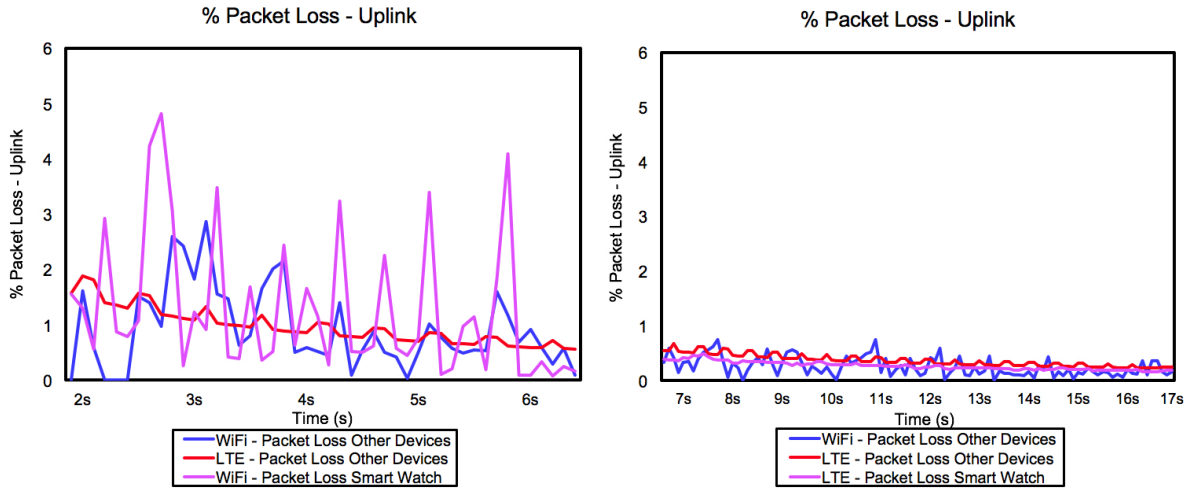
(a) (b)

Figure A.15: High load 22 devices - Delay results for downlink (ms)



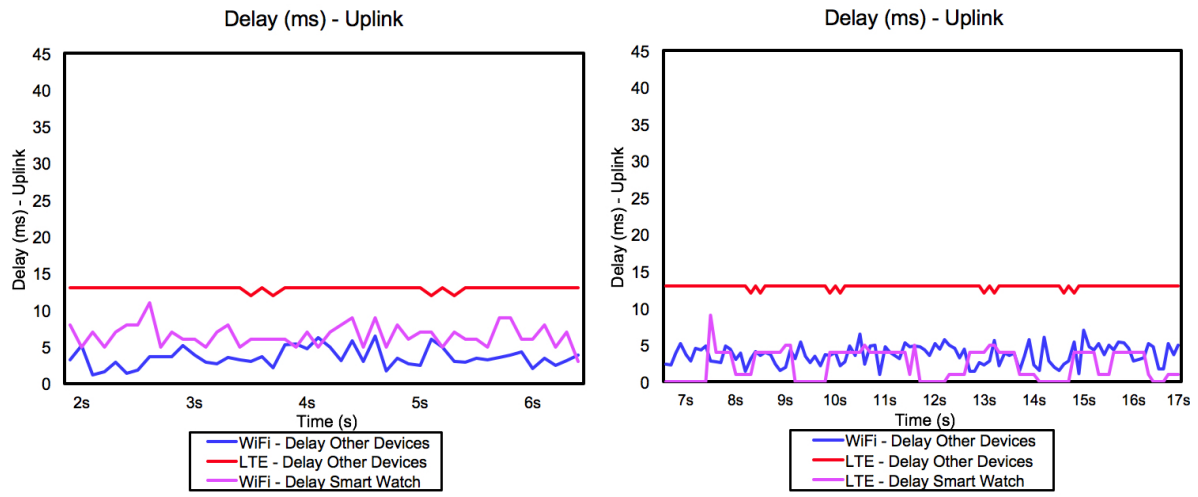
(a) (b)

Figure A.16: High load 22 devices - Max. throughput results for uplink (%)



(a) (b)

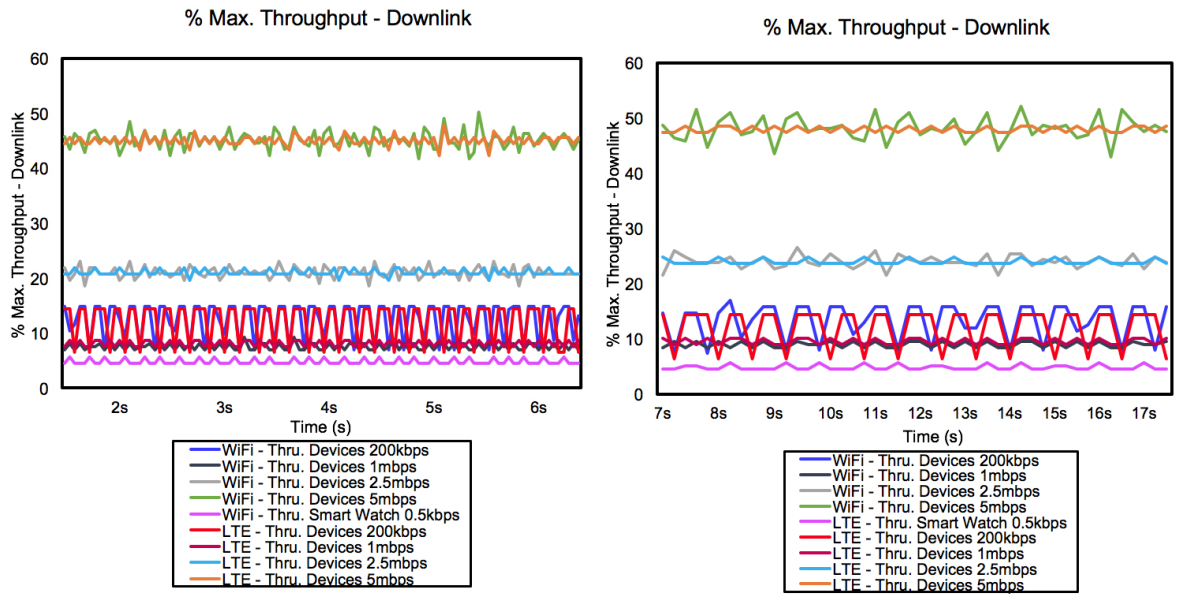
Figure A.17: High load 22 devices - Packet loss results for uplink (%)



(a) (b)

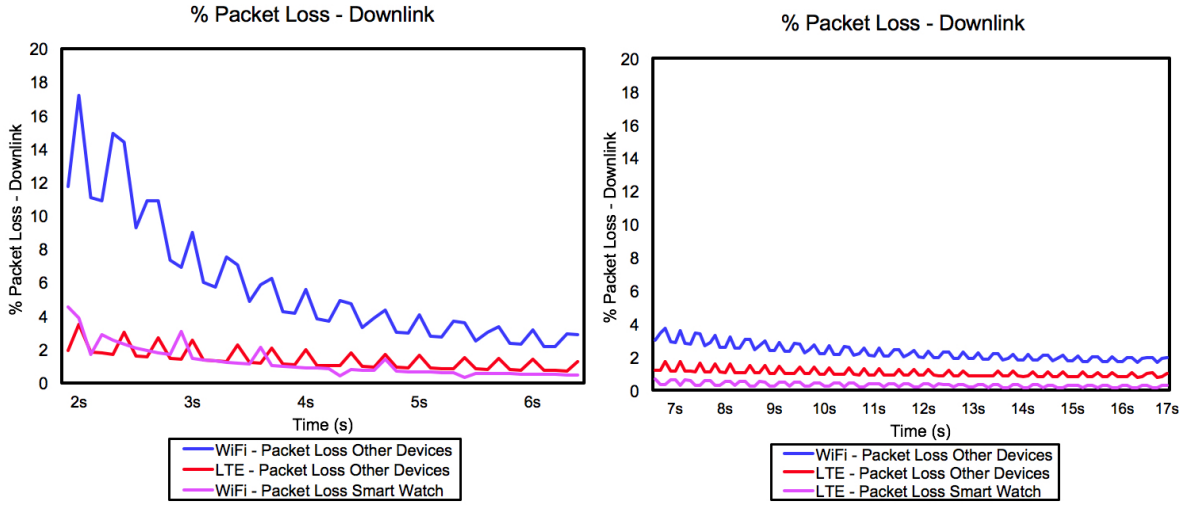
Figure A.18: High load 22 devices - Delay results for uplink (ms)

A.4 REMOS-IoT Scenario 4



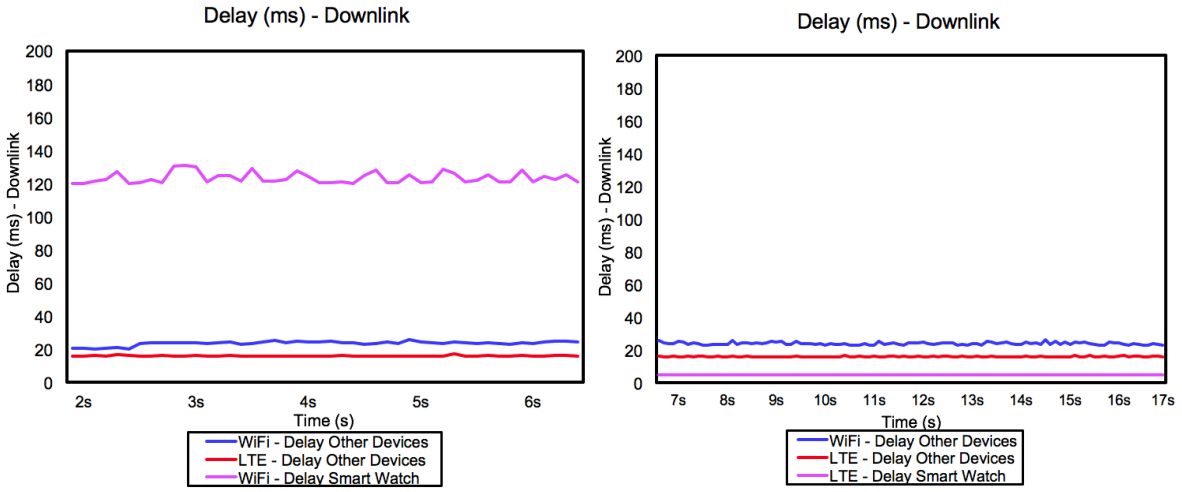
(a) (b)

Figure A.19: Low load 44 devices - Max. throughput results for downlink (%)



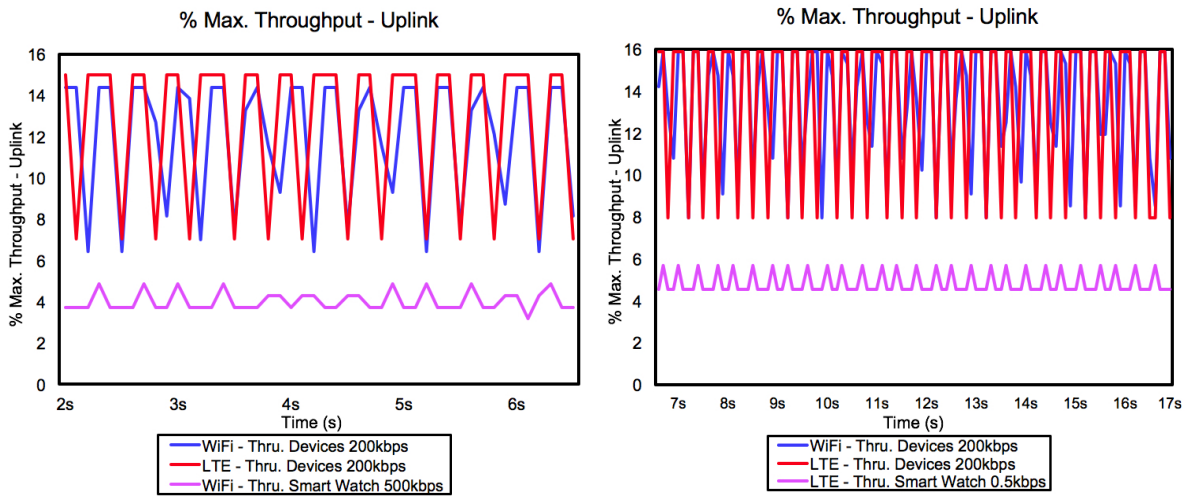
(a) (b)

Figure A.20: Low load 44 devices - Packet loss results for downlink (%)



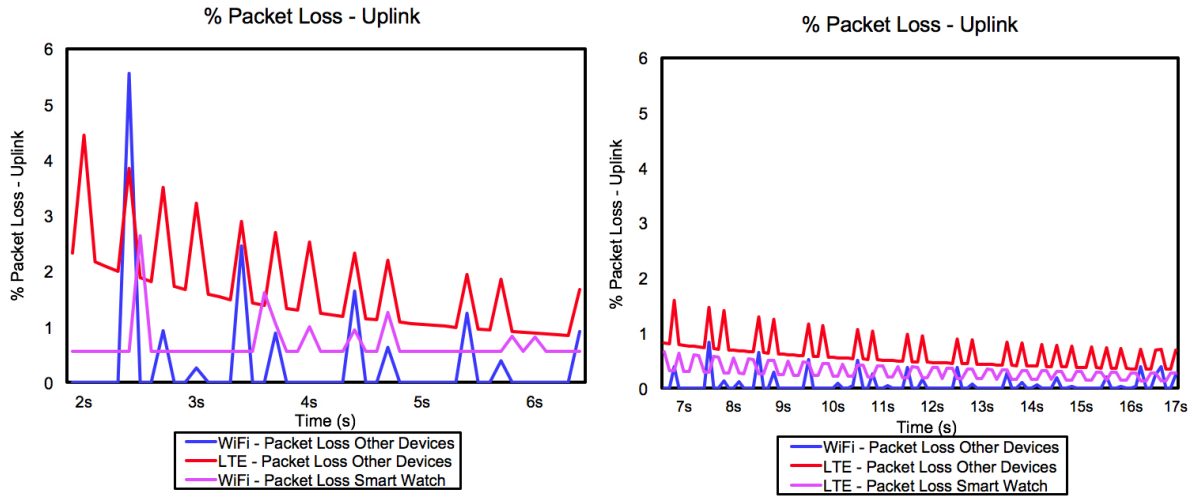
(a) (b)

Figure A.21: Low load 44 devices - Delay results for downlink (ms)



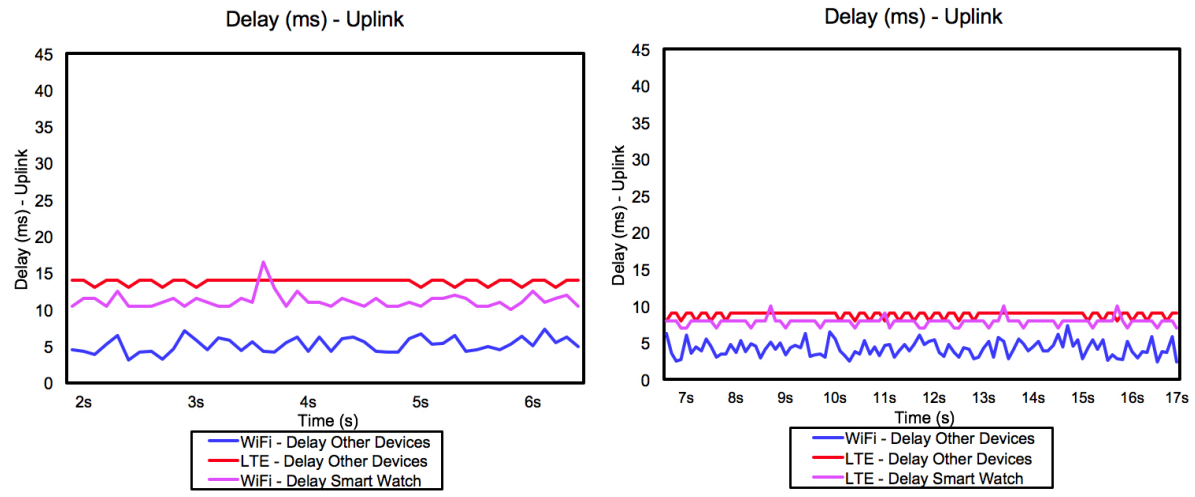
(a) (b)

Figure A.22: Low load 44 devices - Max. throughput results for uplink (%)



(a) (b)

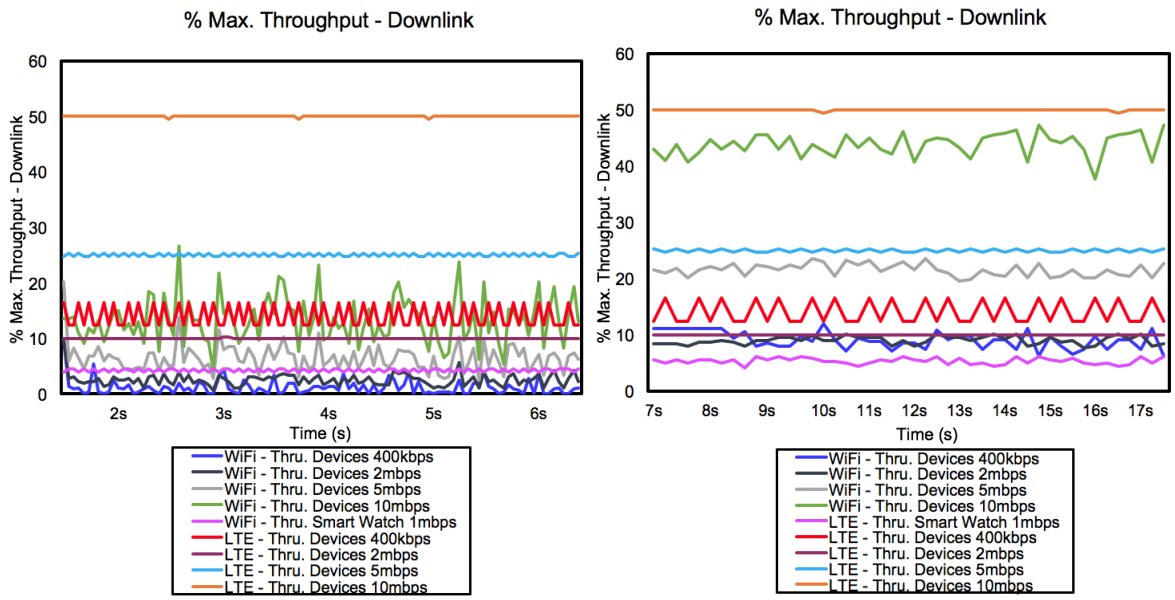
Figure A.23: Low load 44 devices - Packet loss results for uplink (%)



(a) (b)

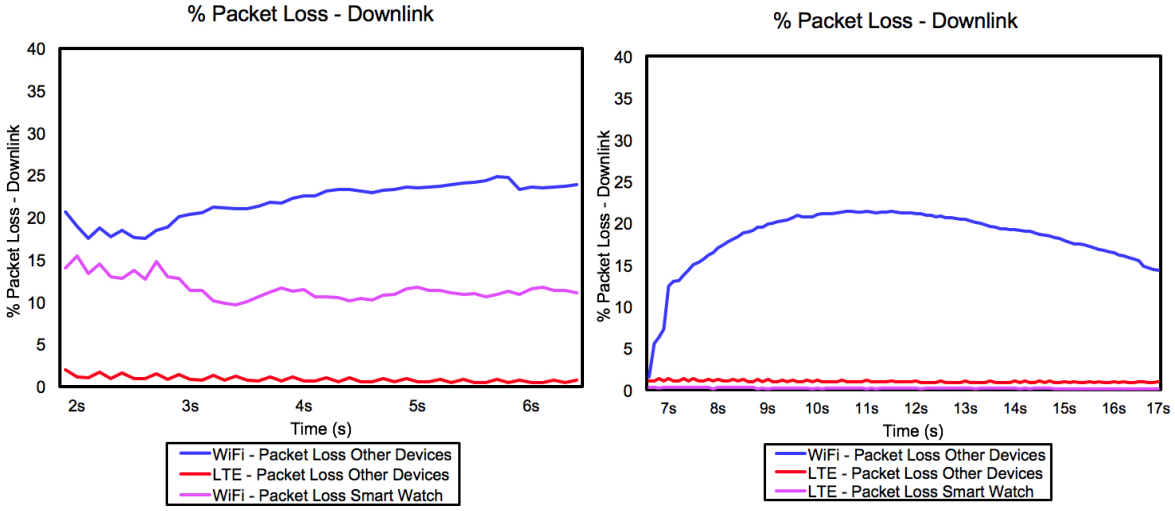
Figure A.24: Low load 44 devices - Delay results for uplink (ms)

A.5 REMOS-IoT Scenario 5



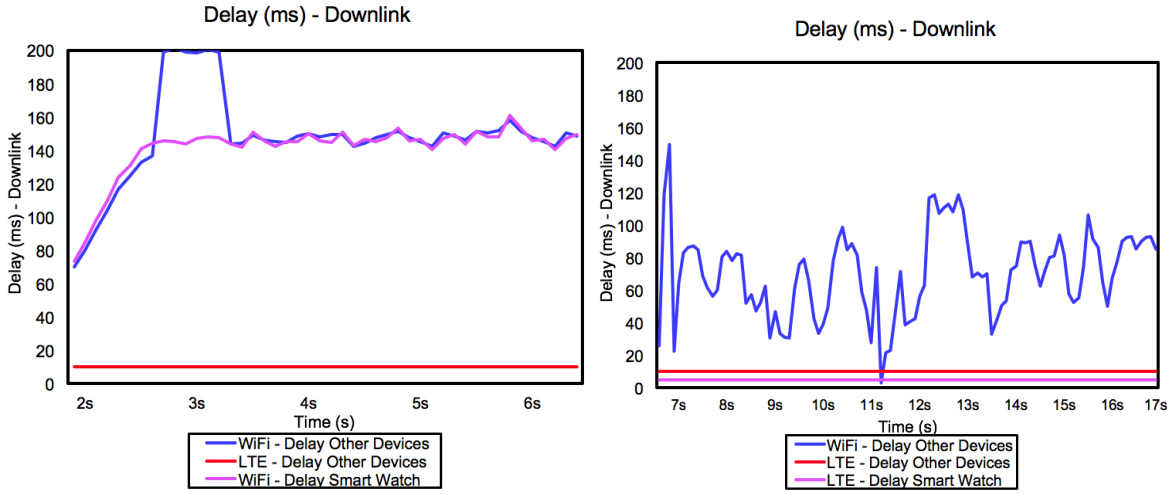
(a) (b)

Figure A.25: Average load 44 devices - Max. throughput results for downlink (%)



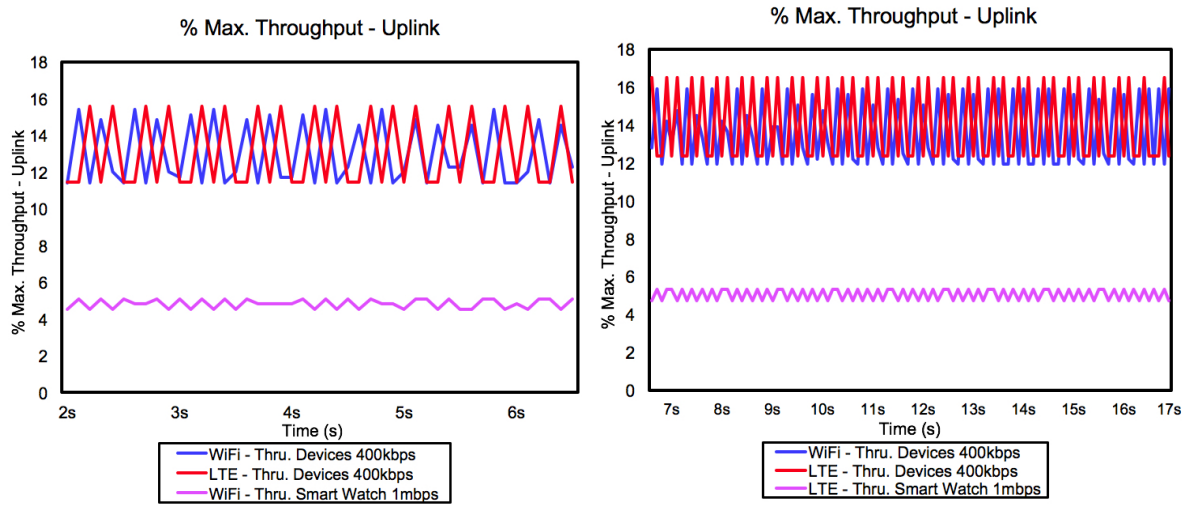
(a) (b)

Figure A.26: Average load 44 devices - Packet loss results for downlink (%)



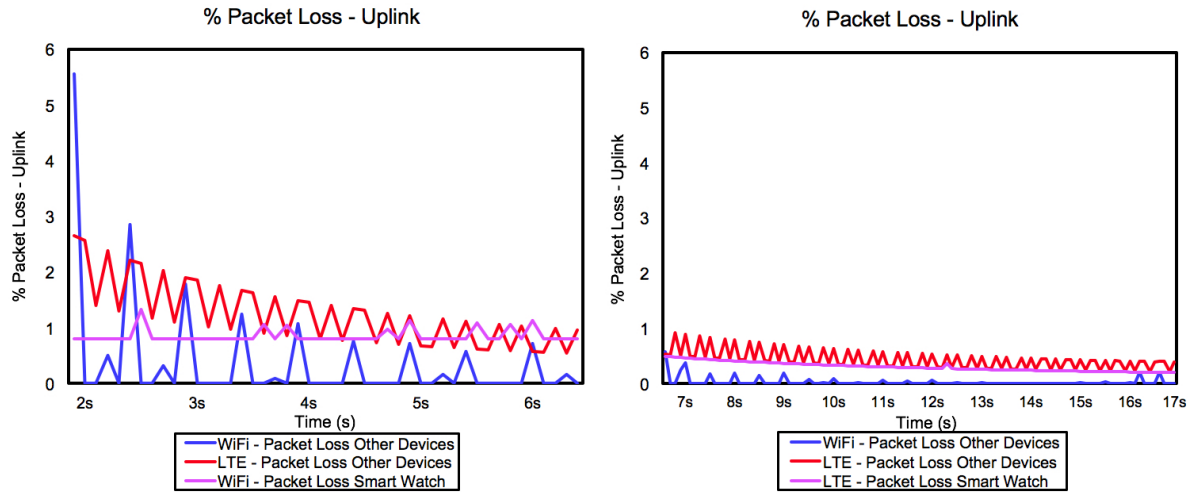
(a) (b)

Figure A.27: Average load 44 devices - Delay results for downlink (ms)



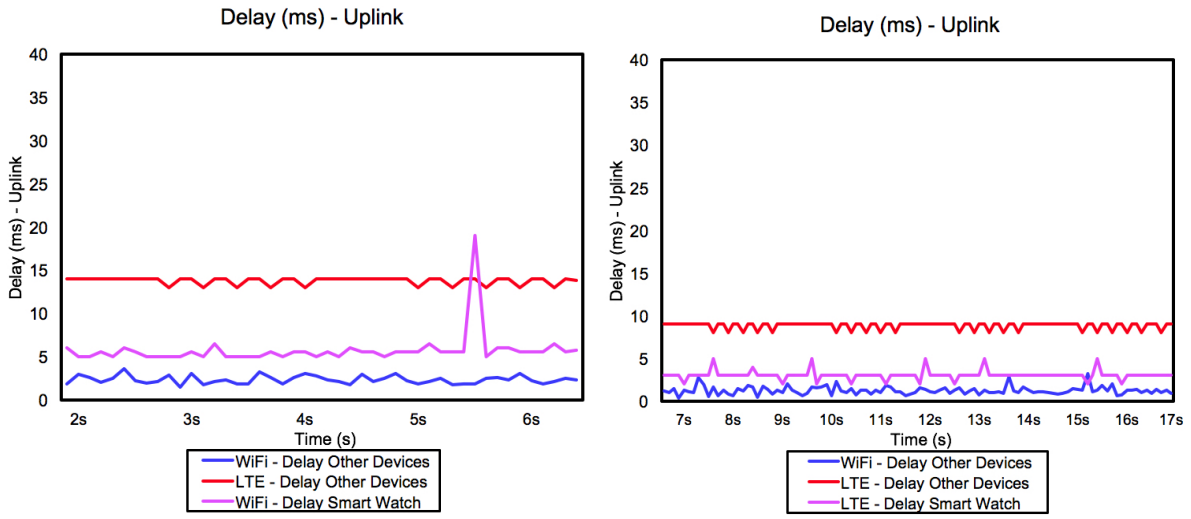
(a) (b)

Figure A.28: Average load 44 devices - Max. throughput results for uplink (%)



(a) (b)

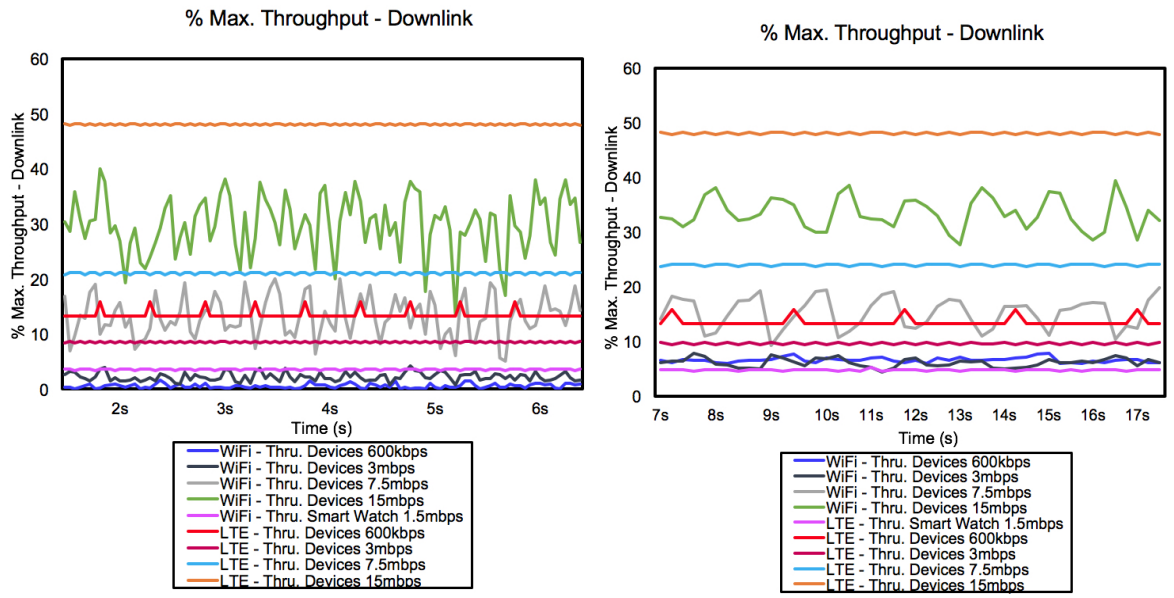
Figure A.29: Average load 44 devices - Packet loss results for uplink (%)



(a) (b)

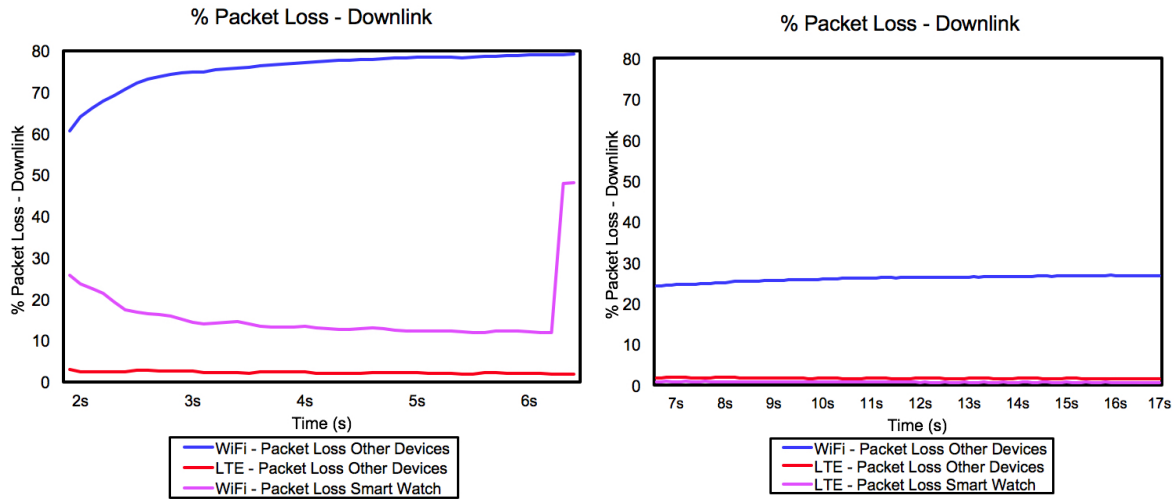
Figure A.30: Average load 44 devices - Delay results for uplink (ms)

A.6 REMOS-IoT Scenario 6



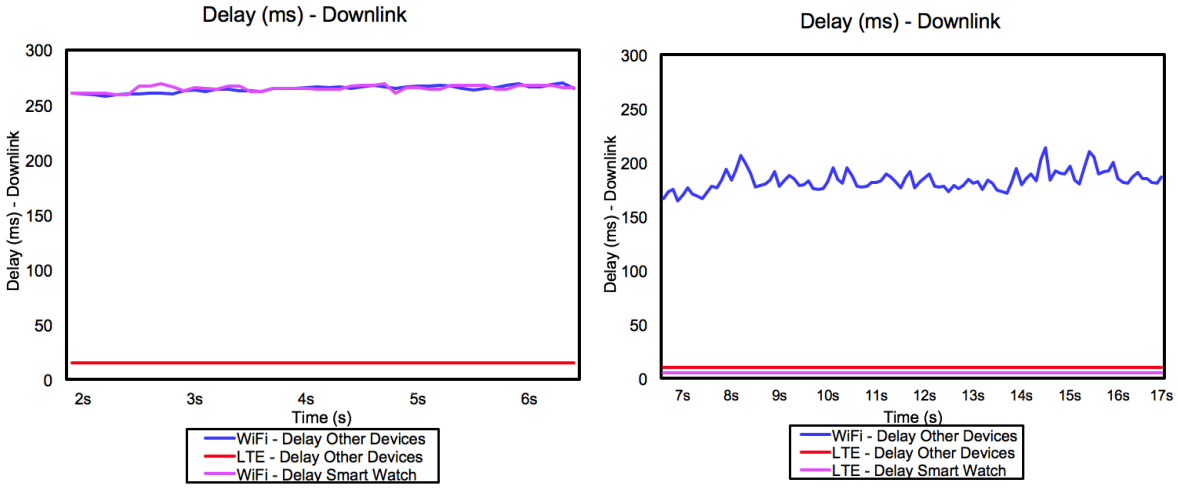
(a) (b)

Figure A.31: High load 44 devices - Max. throughput results for downlink (%)



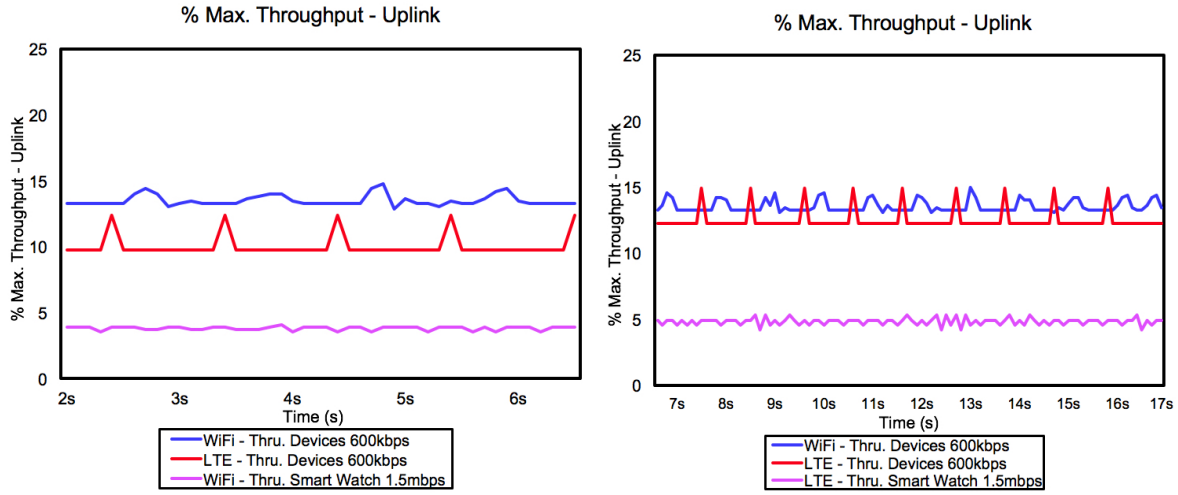
(a) (b)

Figure A.32: High load 44 devices - Packet loss results for downlink (%)



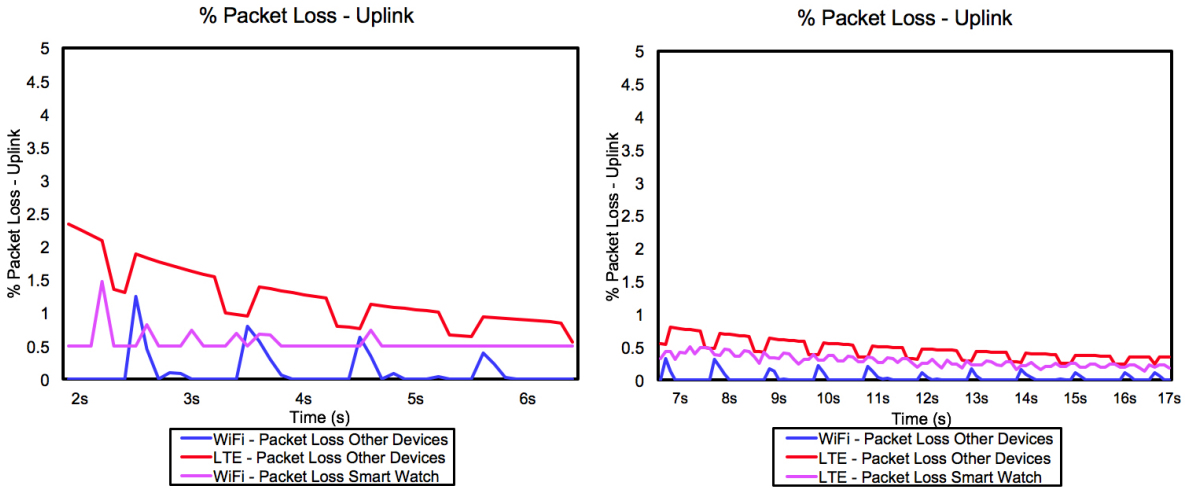
(a) (b)

Figure A.33: High load 44 devices - Delay results for downlink (ms)



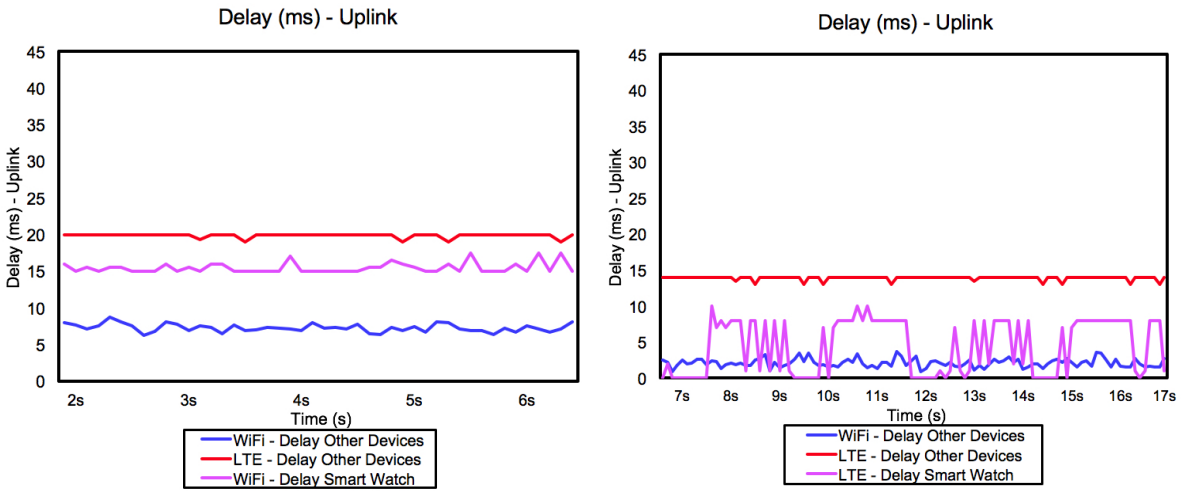
(a) (b)

Figure A.34: High load 44 devices - Max. throughput results for uplink (%)



(a) (b)

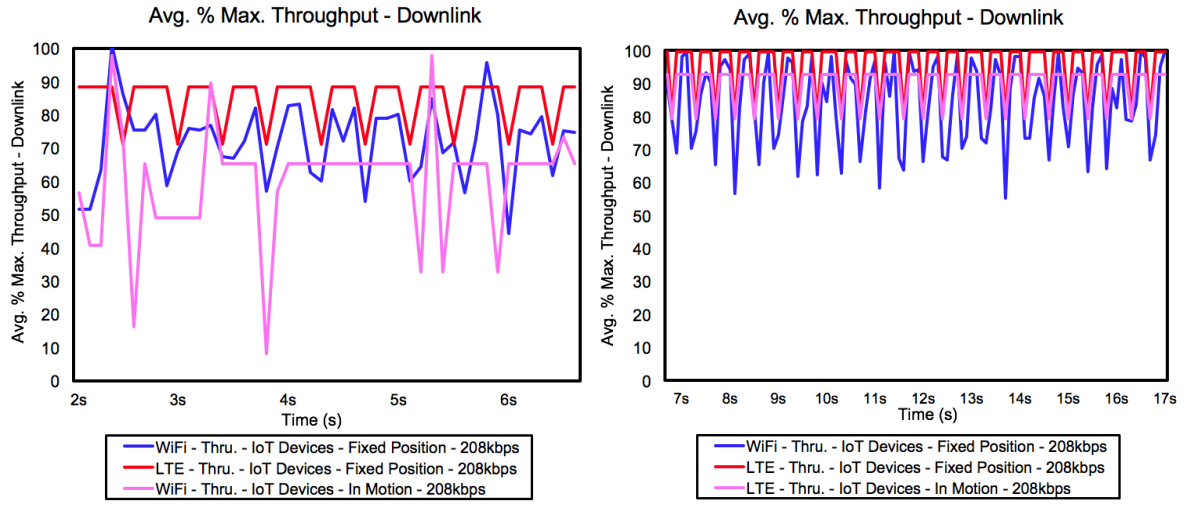
Figure A.35: High load 44 devices - Packet loss results for uplink (%)



(a) (b)

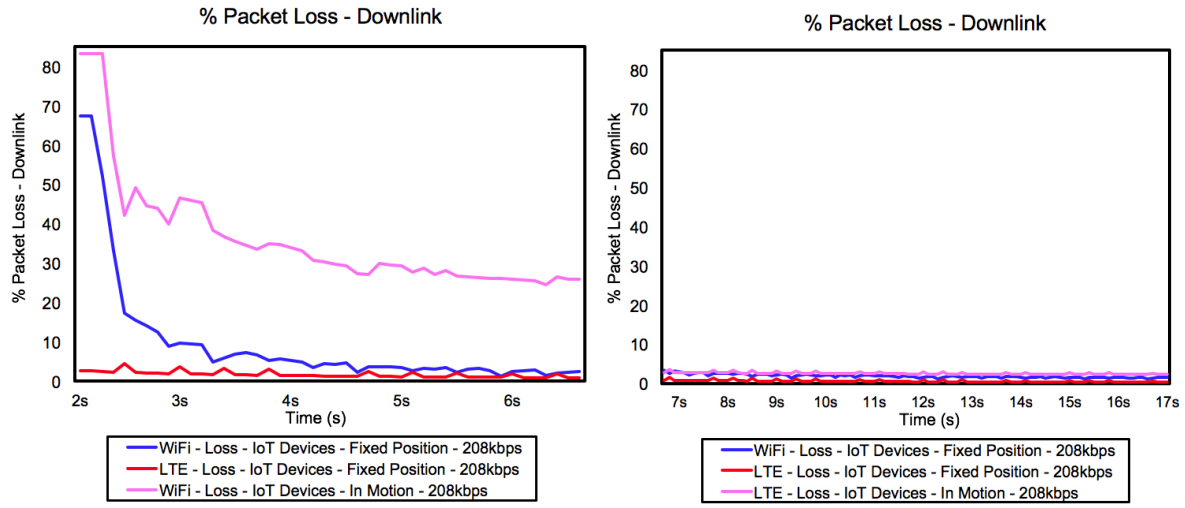
Figure A.36: High load 44 devices - Delay results for uplink (ms)

A.7 REMOS-IoT Scenario 7



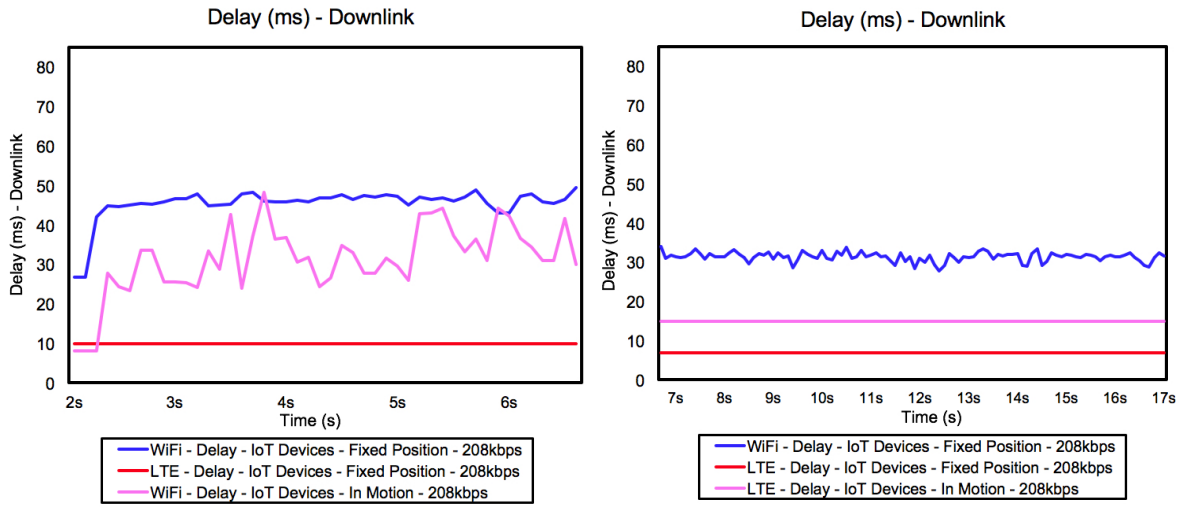
(a) (b)

Figure A.37: Low load 200 devices - Max. throughput results for downlink (%)



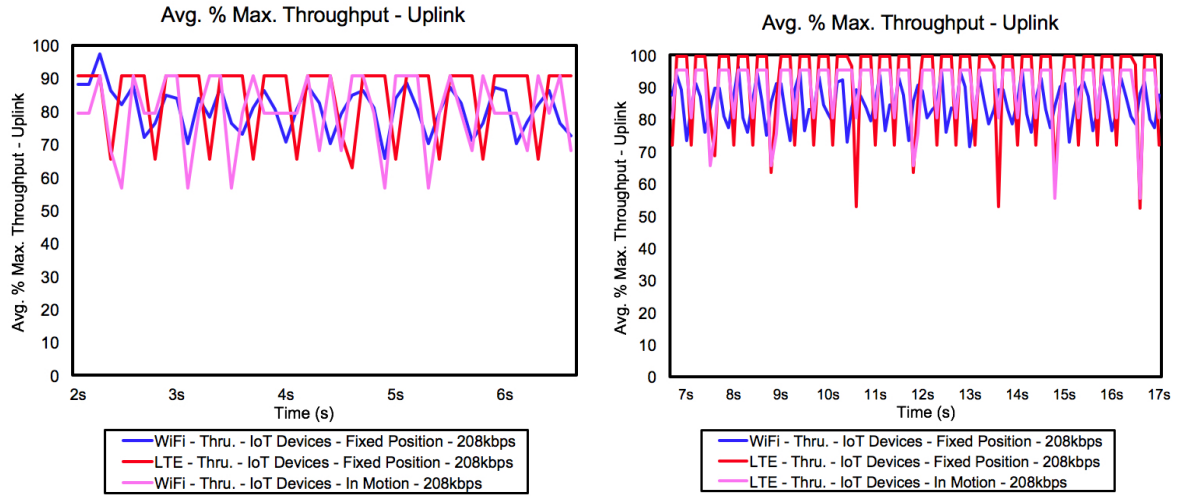
(a) (b)

Figure A.38: Low load 200 devices - Packet loss results for downlink (%)



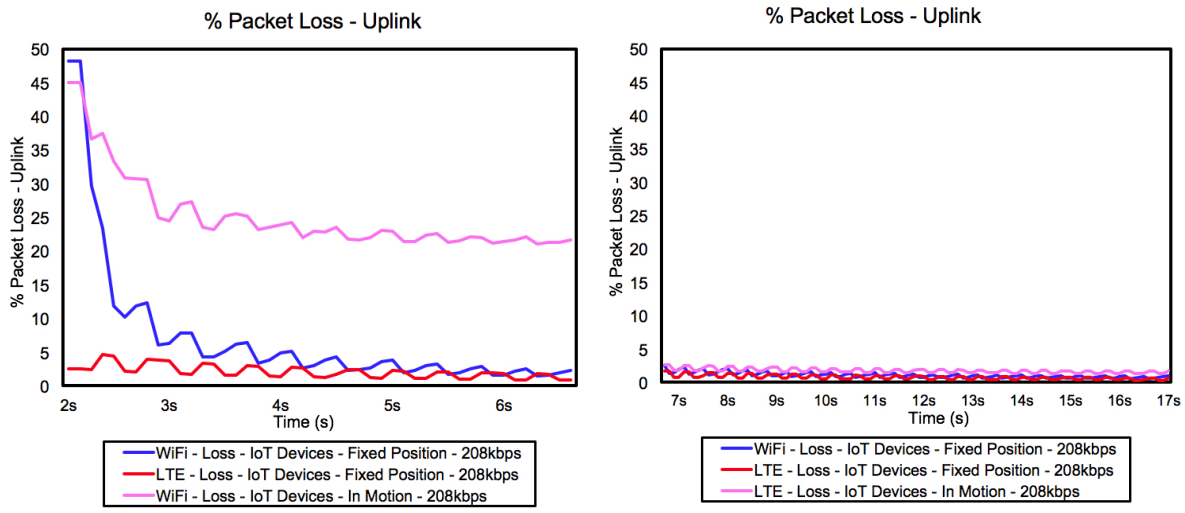
(a) (b)

Figure A.39: Low load 200 devices - Delay results for downlink (ms)



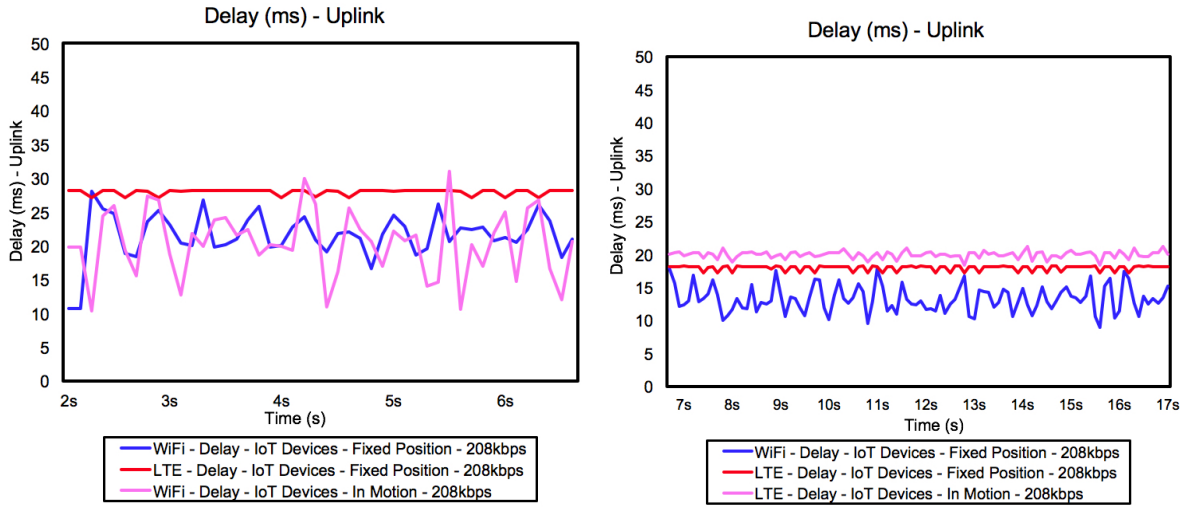
(a) (b)

Figure A.40: Low load 200 devices - Max. throughput results for uplink (%)



(a) (b)

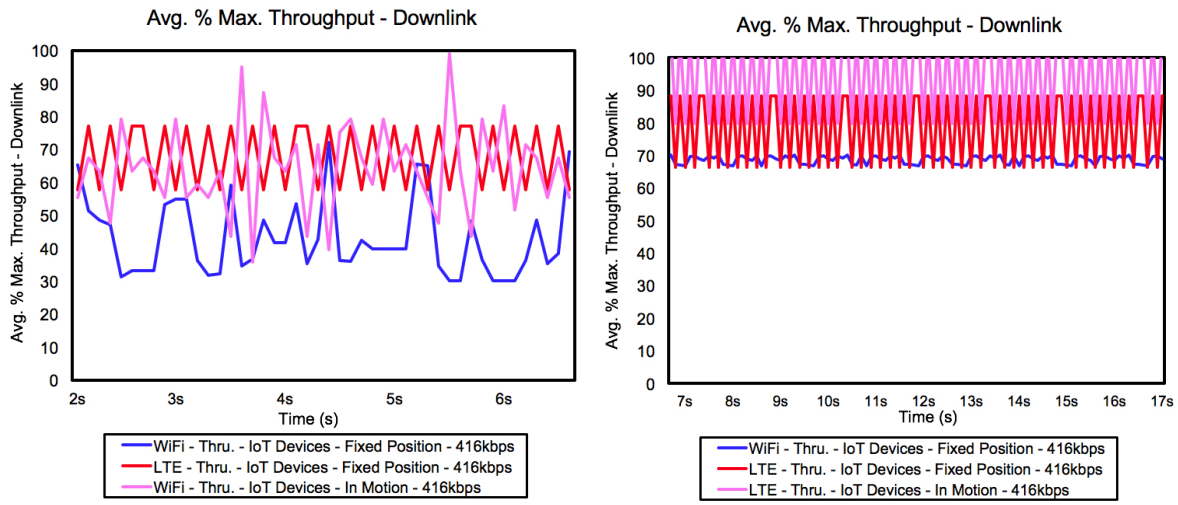
Figure A.41: Low load 200 devices - Packet loss results for uplink (%)



(a) (b)

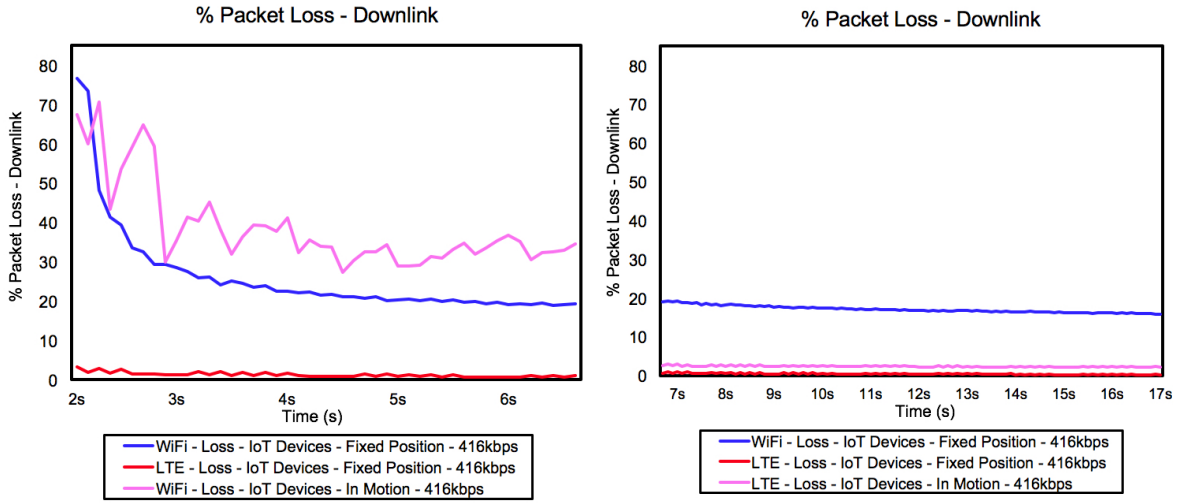
Figure A.42: Low load 200 devices - Delay results for uplink (ms)

A.8 REMOS-IoT Scenario 8



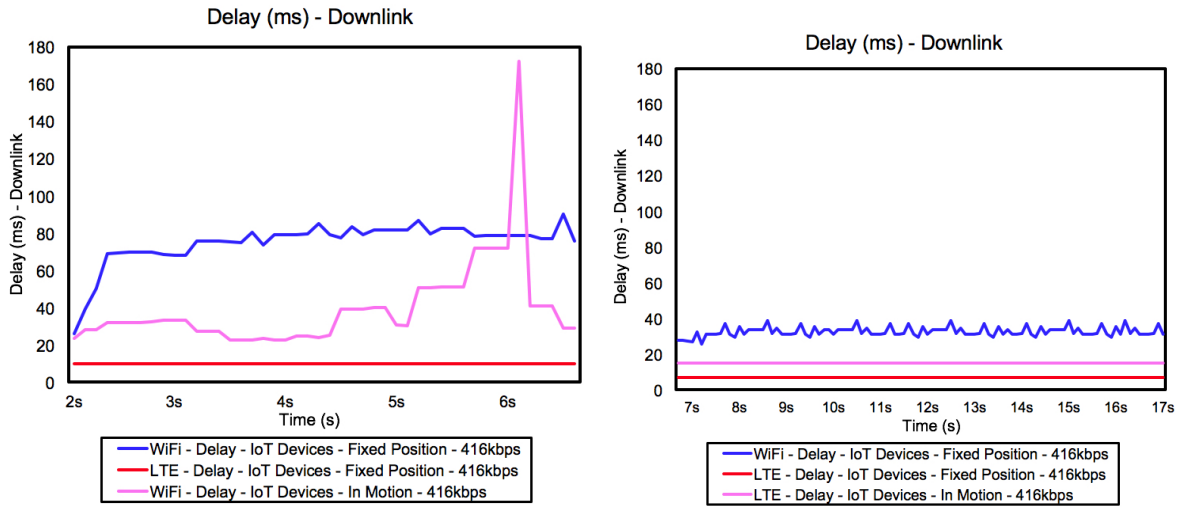
(a) (b)

Figure A.43: Average load 200 devices - Max. throughput results for downlink (%)



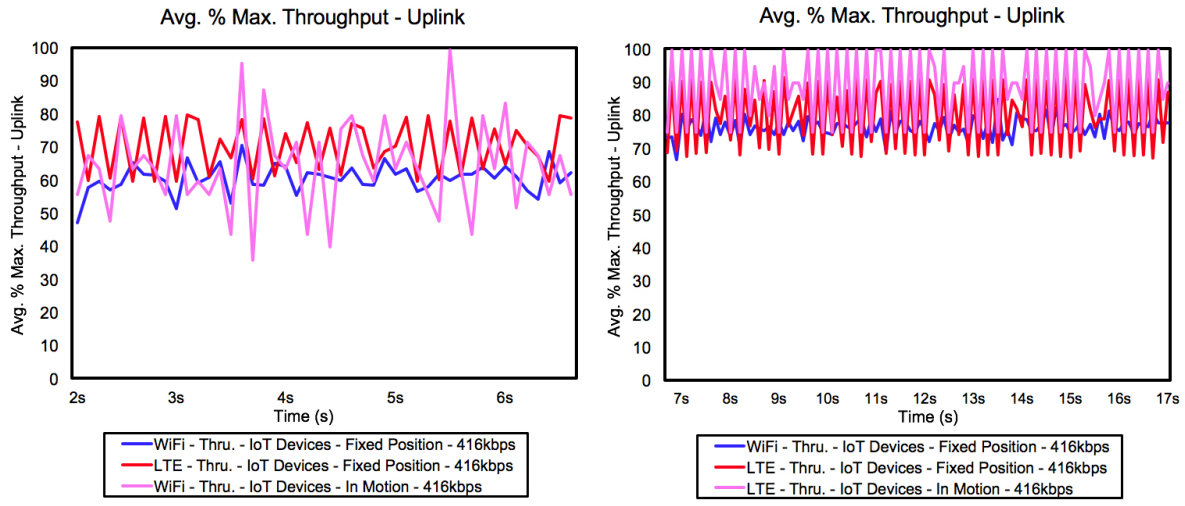
(a) (b)

Figure A.44: Average load 200 devices - Packet loss results for downlink (%)



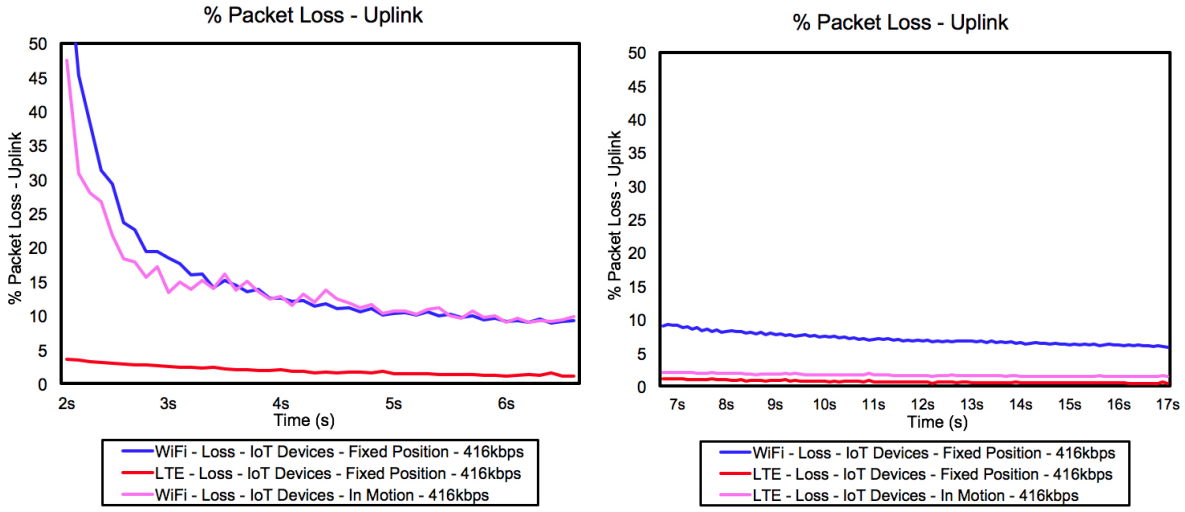
(a) (b)

Figure A.45: Average load 200 devices - Delay results for downlink (ms)



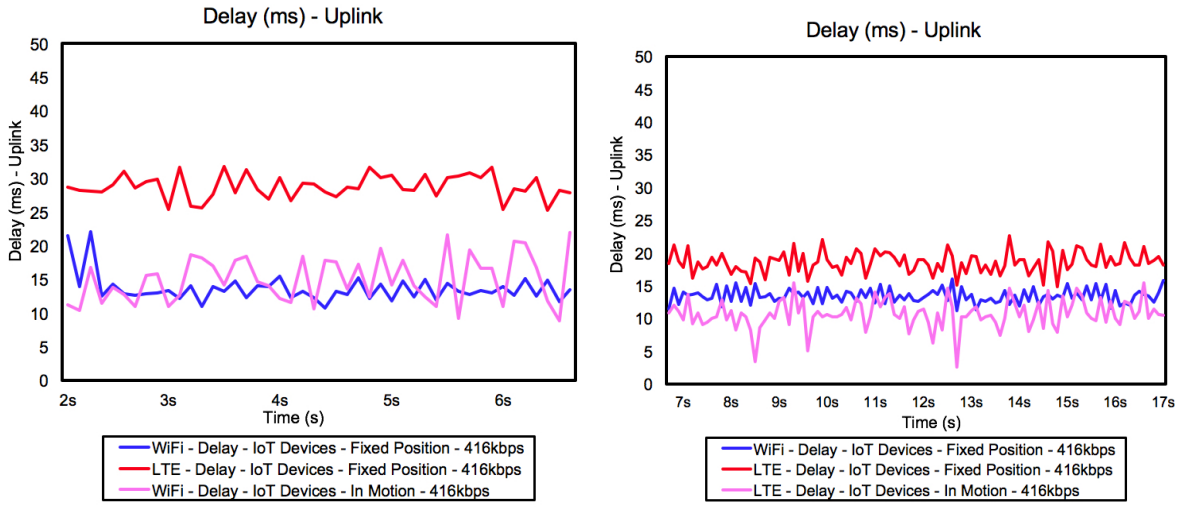
(a) (b)

Figure A.46: Average load 200 devices - Max. throughput results for uplink (%)



(a) (b)

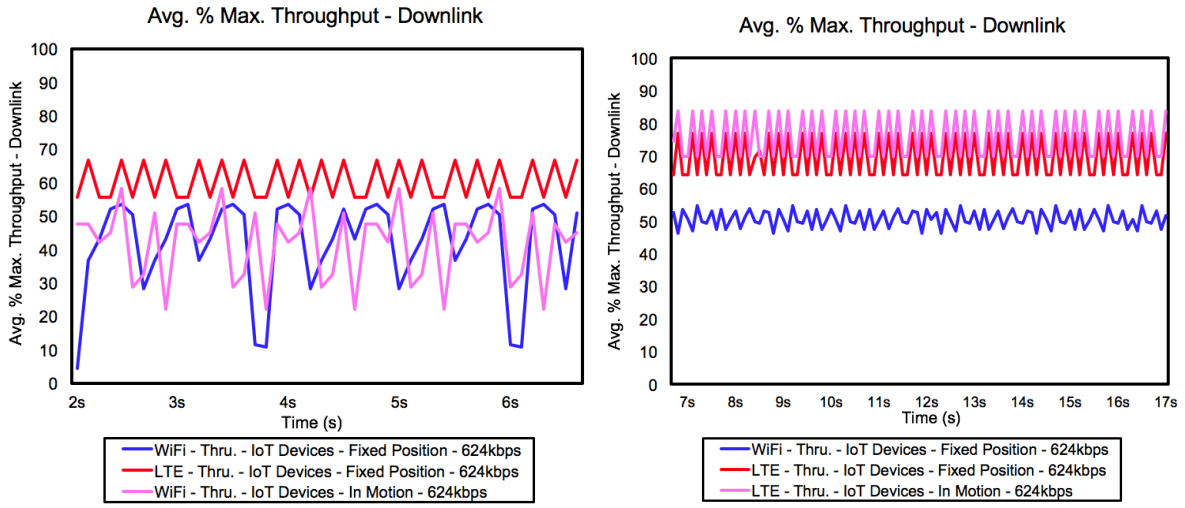
Figure A.47: Average load 200 devices - Packet loss results for uplink (%)



(a) (b)

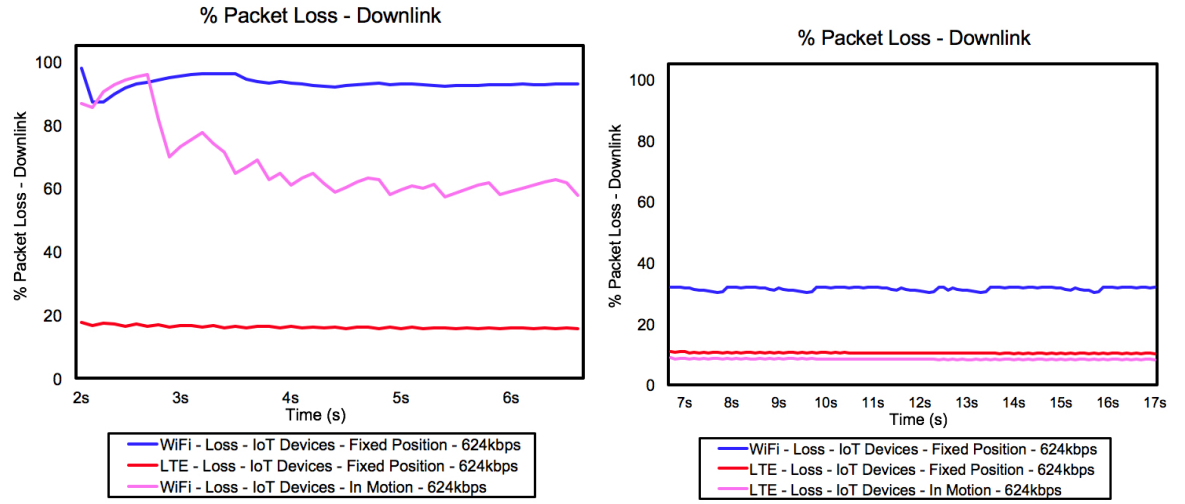
Figure A.48: Average load 200 devices - Delay results for uplink (ms)

A.9 REMOS-IoT Scenario 9



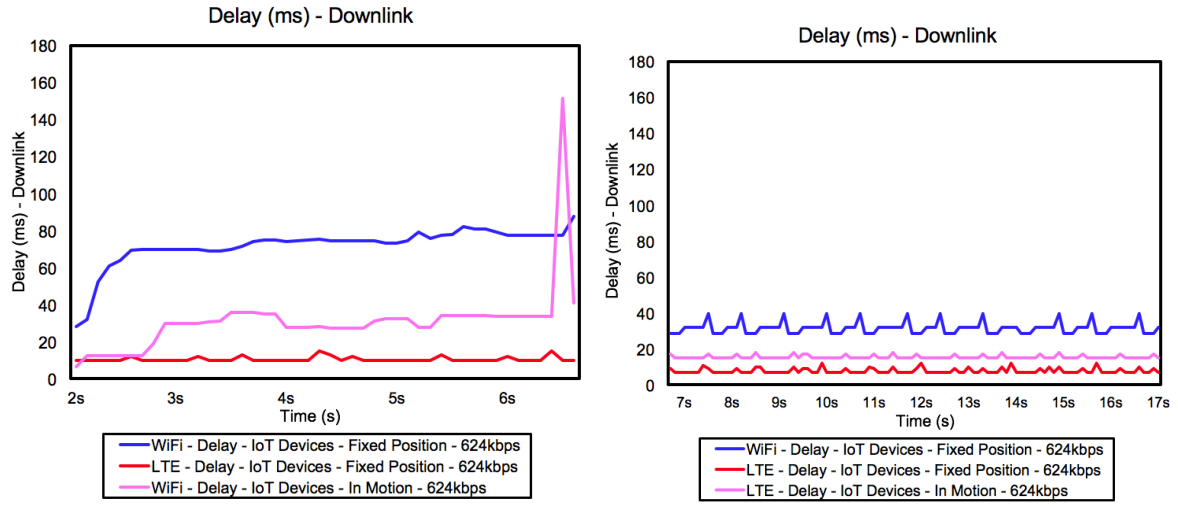
(a) (b)

Figure A.49: High load 200 devices - Max. throughput results for downlink (%)



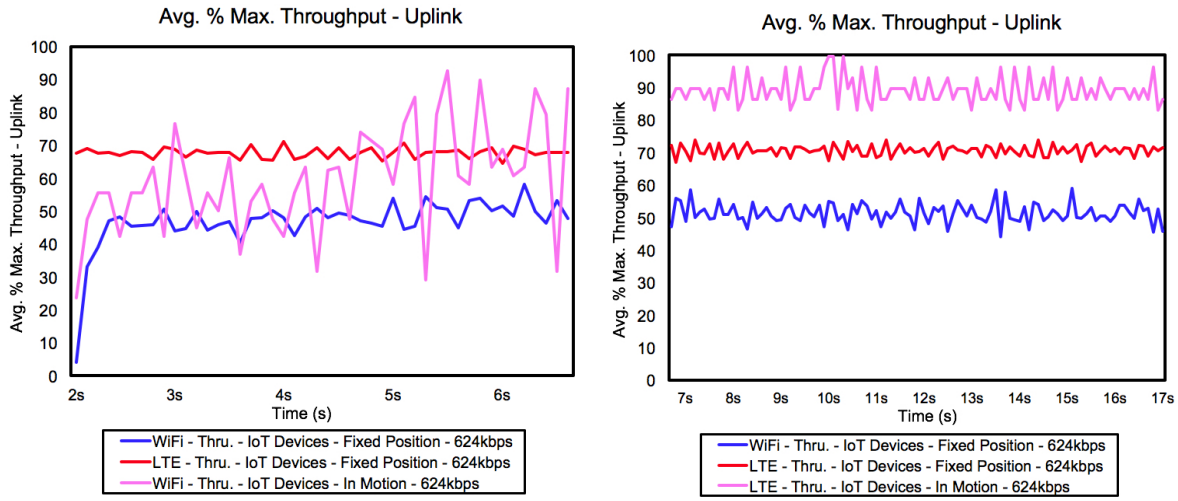
(a) (b)

Figure A.50: High load 200 devices - Packet loss results for downlink (%)



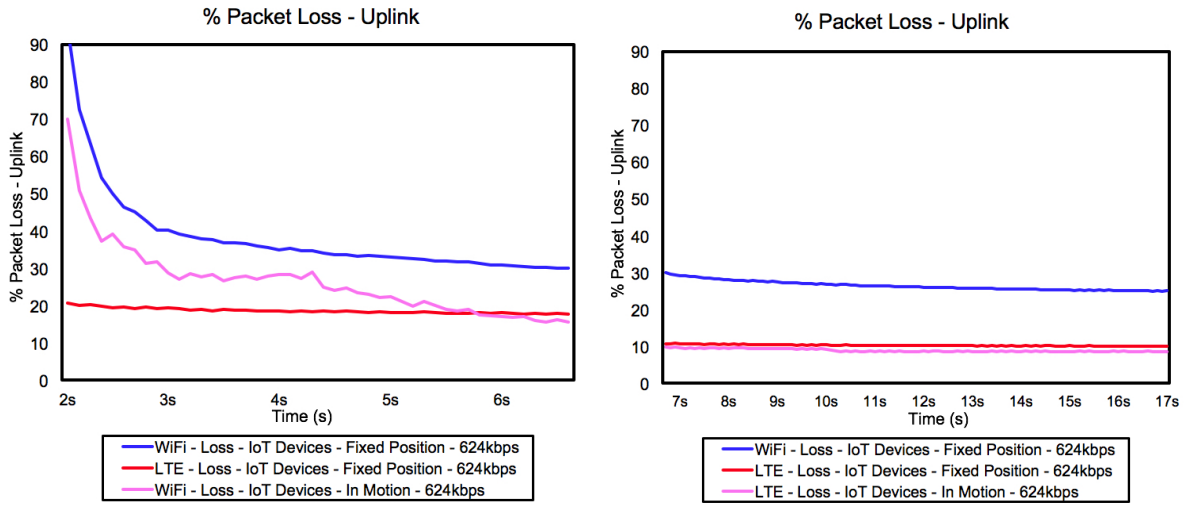
(a) (b)

Figure A.51: High load 200 devices - Delay results for downlink (ms)



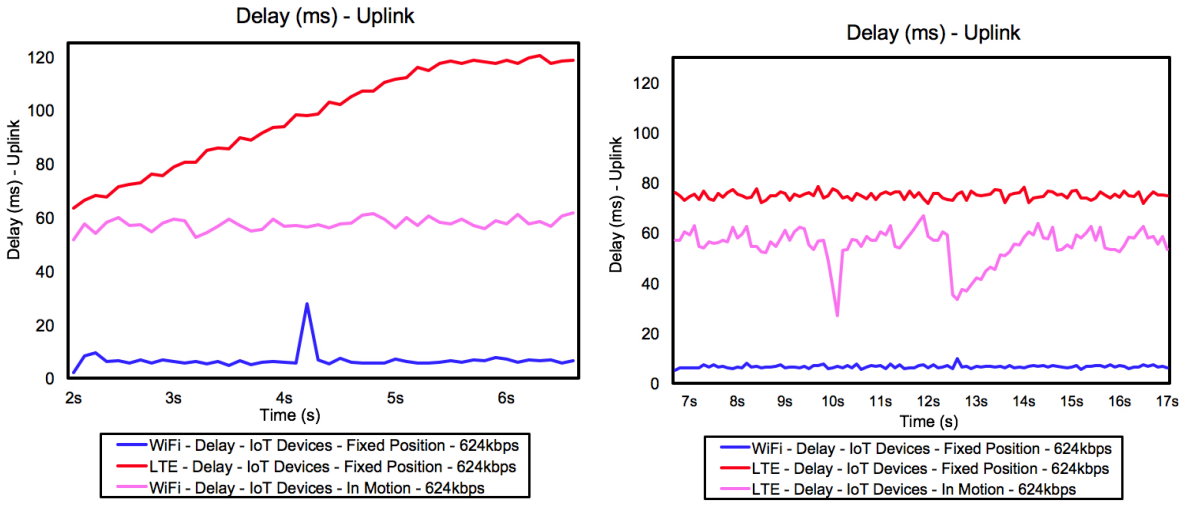
(a) (b)

Figure A.52: High load 200 devices - Max. throughput results for uplink (%)



(a) (b)

Figure A.53: High load 200 devices - Packet loss results for uplink (%)



(a) (b)

Figure A.54: High load 200 devices - Delay results for uplink (ms)

Appendix B

The Twitter Stream Filtering and Tweet Processing algorithms presented in this appendix are used in the social IoT solution presented in chapter 7.5. These algorithms select and verify user tweets with commands to the IoT devices.

B.1 Twitter Stream Filtering Algorithm

The Twitter Stream Filtering algorithm, presented in Algorithm B.1, contains the tasks of connecting to Twitter and setting up a stream (TwitterStreamer()), described in detail in chapter 7, section 7.5.2.1. In order to avoid delays in searching for relevant tweets with the correct hashtag, the TwitterStreamer() is threaded and the tweets are put into a Queue().

Algorithm B.1 Twitter Stream Filtering

```
Track_term = '#GM22FYP__'
Class TwitterStreamer(TwythonStreamer):
        def __init__(Oauth_tokens, Oath_secret, queue):
                super(TwitterStreamer).__init__(Oauth_tokens, Oath_secret, queue)
        def on_success(data):
                if 'text' in data: put_data_in_queue()
        def on_error(data): print data
def receive_tweets(queue):
        TwitterStreamer(Oauth_tokens, Oath_secret, queue)
        TwitterStreamer.statuses.filter(track=Track_term, language=English)
def handle_tweets(tweet_queue):
        while true:
                get_from_queue()
                get_map_from_data()
                if (all_keys_present()): send_data()
def main:
        Thread(target=receive_tweets,args=[Queue()]).start()
```

B.2 Tweet Processing Algorithm

The Tweet Processing algorithm, introduced in Algorithm B.2, performs tweet content validation before data is sent to Adafruit, so it receives data in the format it expects, in a JSON map, as described in chapter 7, section 7.5.2.2.

Algorithm B.2 Tweet Processing

```
def analyse_tweet(tweet):
        try:
                get_map(tweet)
        catch IndexError:
                print(error)
        try:
                convert_to_json(map)
        catch ValueError:
                print(error)
        if(keys "PI", "Component", "Status" are present)
                send_data()
def get_map(tweet)
        return(data.split('{')[1].split('}')[0])
def convert_to_json(map):
        return json.loads(map)
```

Bibliography

[1] G. Sachs, *The Internet of Things: Making sense of the next mega-trend - Goldman Sachs*, 2014. [Online]. Available: https://www.goldmansachs.com/insights/pages/internet-of-things/iot-report.pdf.

[2] P. Newman, *Internet of Things Report: Technology Trends & Market Growth in 2019 - Business Insider*, 2019. [Online]. Available: https://www.businessinsider.com/internet-of-things-report?r=US%7B%5C&%7DIR=T.

[3] P. Biggs, J. Garrity, C. LaSalle, A. Polomska and R. Pepper, *Harnessing the Internet of Things for Global Development - Cisco and ITU*, 2016. [Online]. Available: https://www.itu.int/en/action/broadband/Documents/Harnessing-IoT-Global-Development.pdf.

[4] Axelerate, *Get the IoT to work*, 2017. [Online]. Available: http://www.axeleratesolutions.com/services/iot-platform/.

[5] D. Sinh, L. V. Le, B. S. P. Lin and L. P. Tung, 'SDN/NFV - A new approach of deploying network infrastructure for IoT', in *27th Wireless and Optical Communication Conference (WOCC)*, 2018, pp. 1–5.

[6] N. Narendra and P. Misra, *Research Challenges in the Internet of Mobile Things*, 2016. [Online]. Available: https://iot.ieee.org/newsletter/march-2016/research-challenges-in-the-internet-of-mobile-things.html.

[7] C. Rowland, *What's Different About User Experience Design for the Internet of Things?*, 2015. [Online]. Available: https://uxmag.com/articles/whats-different-about-user-experience-design-for-the-internet-of-things.

[8] C. Peng, X. Tan, M. Gao and Y. Yao, 'Virtual Reality in Smart City', in *Geo-Informatics in Resource Management and Sustainable Ecosystem. Communications in Computer and Information Science*, vol. 398, Springer, Berlin, Heidelberg, 2013, pp. 107–118.

[9] *The NS-3 Network Simulator*, 2019. [Online]. Available: https://www.nsnam.org/.

[10] E. Y. Song, G. J. FitzPatrick and K. B. Lee, 'Smart Sensors and Standard-based Interoperability in Smart Grids', *IEEE Sensors Journal*, vol. 1748, no. c, pp. 1–1, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7986956/.

[11] A. Akinsiku and D. Jadav, 'BeaSmart: A beacon enabled smarter workplace', *Proceedings of the NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pp. 1269–1272, 2016.

[12] C. Julien, C. Liu, A. L. Murphy and G. P. Picco, 'BLEnd: Practical Continuous Neighbor Discovery for Bluetooth Low Energy', *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pp. 105–116, 2017.

[13] W. Hu, W. Zhang, H. Hu, Y. Wen and K.-J. Tseng, 'Toward Joint Compression – Transmission Optimization for Green Wearable Devices: An Energy-Delay Tradeoff', *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 1006–1018, 2017.

[14] I. Pavic, J. Puskar, I. Soldo, I. Spasic, H. Dzapo and D. Cika, 'Building network-enabled smart sensors and actuators', *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2015 - Proceedings*, no. May, pp. 1597–1601, 2015.

[15] C. Nan, Y. Lee, F. Tila, S. Lee and D. H. Kim, 'A study of actuator network middleware based on ID for IoT system', *Proceedings - 8th International Conference on Grid and Distributed Computing, GDC 2015*, pp. 17–19, 2016.

[16] R. Parada and J. Melia-Segui, 'Gesture Detection Using Passive RFID Tags to Enable People-Centric IoT Applications', *IEEE Communications Magazine*, vol. 55, no. 2, pp. 56–61, 2017.

[17] E. Fernandes, A. Rahmati, J. Jung and A. Prakash, 'Security Implications of Permission Models in Smart-Home Application Frameworks', *IEEE Security and Privacy*, vol. 15, no. 2, pp. 24–30, 2017.

[18] B. Carnevale, L. Baldanzi, L. Pilato and L. Fanucci, 'A flexible System-on-a-Chip implementation of the Advanced Encryption Standard', *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, pp. 156–161, 2016.

[19] Y. Park, S. Kuk, I. Kang and H. Kim, 'Overcoming IoT Language Barriers Using Smartphone SDRs', *IEEE Transactions on Mobile Computing*, vol. 16, no. 3, pp. 816–828, 2017.

[20] U. A. Noman, B. Negash, A. M. Rahmani, P. Liljeberg and H. Tenhunen, 'From threads to events: Adapting a lightweight middleware for Contiki OS', *2017 14th IEEE Annual Consumer Communications and Networking Conference, CCNC 2017*, pp. 486–491, 2017.

[21] F. Tong, R. Zhang and J. Pan, 'One Handshake Can Achieve More: An Energy-Efficient, Practical Pipelined Data Collection for Duty-Cycled Sensor Networks', *IEEE Sensors Journal*, vol. 16, no. 9, pp. 3308–3322, 2016.

[22] *The NS-3 network simulator*, 2018. [Online]. Available: https://www.nsnam.org/.

[23] S. G. Prasad, R. Vivek, J. Mungara and E. J. Sebastian, 'NS3 simulation studies for optimized neighbour discovery in 6LoWPAN networks', *2016 IEEE 3rd International Symposium on Wireless Systems within the IEEE International Conferences on Intelligent Data Acquisition and Advanced Computing Systems, IDAACS-SWS 2016 - Proceedings*, no. September, pp. 15–18, 2017.

[24] L. Tian, J. Famaey and S. Latr, 'Evaluation of the IEEE 802 . 11ah Restricted Access Window Mechanism for dense IoT networks Evaluation of the IEEE 802 . 11ah Restricted Access Window Mechanism for dense IoT networks', *IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, no. May, pp. 1–9, 2016.

[25] T. V. Chien, H. N. Chan and T. N. Huu, 'A comparative study on operating system for Wireless Sensor Networks', *2011 International Conference on Advanced Computer Science and Information Systems*, pp. 73–78, 2011.

[26] W. Shang, A. Afanasyev and L. Zhang, 'The design and implementation of the NDN protocol stack for RIOT-OS', *2016 IEEE Globecom Workshops, GC Wkshps 2016 - Proceedings*, pp. 0–5, 2016.

[27] *Android Developers*, 2018. [Online]. Available: https://developer.android.com/.

[28] R. K. Behera, S. Gupta and A. Gautam, 'Big-data empowered cloud centric Internet of Things', *Proceedings - 2015 International Conference on Man and Machine Interfacing, MAMI 2015*, 2016.

[29] W. Shang, Z. Wang, A. Afanasyev, J. Burke and L. Zhang, 'Breaking out of the Cloud: Local Trust Management and Rendezvous in Named Data Networking of Things', *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation - IoTDI '17*, pp. 3–13, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3054977.3054993.

[30] Z. Wang, K. Jiang, M. Du and J. Le, 'Content-based query for class-level tagged items via the EPC network', *2014 International Conference on the Internet of Things, IOT 2014*, pp. 43–48, 2014.

[31] N. Koshizuka and K. Sakamura, 'Ubiquitous ID: Standards for ubiquitous computing and the internet of things', *IEEE Pervasive Computing*, vol. 9, no. 4, pp. 98–101, 2010.

[32] T. M. Fernandez-Carames and P. Fraga-Lamas, 'A Review on Human-Centered IoT-Connected Smart Labels for the Industry 4.0', *IEEE Access*, vol. 6, pp. 25 939–25 957, 2018.

[33] T. Marktscheffel, W. Gottschlich, W. Popp, P. Werli, S. D. Fink, A. Bilzhause and H. De Meer, 'QR code based mutual authentication protocol for Internet of Things', in *WoWMoM 2016 - 17th International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2016, pp. 1–6.

[34] T. Savolainen, J. Soininen and B. Silverajan, 'IPv6 addressing strategies for IoT', *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3511–3519, 2013.

[35] D. He and S. Zeadally, 'An Analysis of RFID Authentication Schemes for Internet of Things in Healthcare Environment Using Elliptic Curve Cryptography', *IEEE Internet of Things Journal*, vol. 2, no. 1, pp. 72–83, 2015.

[36] T. Leppänen, I. S. Milara, J. Yang, J. Kataja and J. Riekki, 'Enabling user-centered interactions in the Internet of Things', *2016 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2016 - Conference Proceedings*, pp. 1537–1543, 2017.

[37] G. Ranganathan, G. Bögel, F. Meyer and A. Grabmaier, 'A Survey of UWB Technology Within RFID Systems and Wireless Sensor Networks', *Smart SysTech 2016; European Conference on Smart Objects, Systems and Technologies*, pp. 56–62, 2016.

[38] A. F. III, V. Khanna, G. Tuncay, R. Want and R. Kravets, 'Bluetooth Low Energy in Dense IoT Environments', *IEEE Communications Magazine*, vol. 54, no. 12, pp. 30–36, 2016.

[39] C. Zhao, L. Huang, Y. Zhao and X. Du, 'Secure machine-type communications toward LTE heterogeneous networks', *IEEE Wireless Communications*, vol. 24, no. 1, pp. 82–87, 2017.

[40] S. Liu, Y. Liu, F. Wu and W. Fan, 'Feature Data Selection for Improving the Performance of Entity Similarity Searches in the Internet of Things', *IEEE Access*, vol. 7, pp. 49 938–49 944, 2019.

[41] A. Ghosh, A. Maeder, M. Baker and D. Chandramouli, '5G Evolution: A View on 5G Cellular Technology Beyond 3GPP Release 15', *IEEE Access*, vol. 7, no. March, pp. 127 639–127 651, 2019.

[42] S. K. Goudos, M. Deruyck, D. Plets, L. Martens, K. E. Psannis, P. Sarigiannidis and W. Joseph, 'A Novel Design Approach for 5G Massive MIMO and NB-IoT Green Networks Using a Hybrid Jaya-Differential Evolution Algorithm', *IEEE Access*, vol. 7, pp. 105 687–105 700, 2019.

[43] GSMA, 'Mobile IoT in the 5G Future: NB-IoT and LTE-M in the context of 5G', Tech. Rep., 2018. [Online]. Available: https://www.ericsson.com/en/networks/trending/insights-and-reports/nb-iot-and-lte-m-in-the-context-of-5g-industry-white-paper.

[44] M. Chen, Y. Miao, X. Jian, X. Wang and I. Humar, 'Cognitive-LPWAN: Towards Intelligent Wireless Services in Hybrid Low Power Wide Area Networks', *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 2, pp. 409–417, Jun. 2019.

[45] D. Gomez-Barquero, D. Navratil, S. Appleby and M. Stagg, 'Point-to-Multipoint Communication Enablers for the Fifth Generation of Wireless Systems', *IEEE Communications Standards Magazine*, vol. 2, no. 1, pp. 53–59, 2018.

[46] M. Saily, C. Barjau, D. Navratil, A. Prasad, D. Gomez-Barquero and F. Tesema, '5G Radio Access Networks: Enabling Efficient Point-to-Multipoint Transmissions', *IEEE Vehicular Technology Magazine*, 2019.

[47] D. Thomas, R. McPherson, G. Paul and J. Irvine, 'Optimizing Power Consumption of Wi-Fi for IoT Devices: An MSP430 processor and an ESP-03 chip provide a power-efficient solution.', *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 92–100, 2016.

[48] K. Jahed, O. Farhat, G. Al-Jurdi and S. Sharafeddine, 'Optimized group owner selection in WiFi direct networks', *2016 24th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2016*, 2016.

[49] K. Beckmann and O. Dedi, 'sDDS: A portable data distribution service implementation for WSN and IoT platforms', *Intelligent Solutions in Embedded Systems (WISES), 2015 12th International Workshop on*, pp. 115–120, 2015.

[50] A. Betzler, C. Gomez, I. Demirkol and J. Paradells, 'CoAP congestion control for the internet of things', *IEEE Communications Magazine*, vol. 54, no. 7, pp. 154–160, 2016.

[51] Z. B. Babovic, J. Protic and V. Milutinovic, 'Web Performance Evaluation for Internet of Things Applications', *IEEE Access*, vol. 4, pp. 6974–6992, 2016.

[52] S. Sreeraj, N. Suresh Kumar and G. Santhosh Kumar, 'A framework for predicting the performance of IoT protocols, a use case based approach', in *International Conference On Smart Technology for Smart Nation, (SmartTechCon)*, 2017, pp. 577–580.

[53] H. W. Chen and F. J. Lin, 'Converging MQTT resources in ETSI standards based M2M platform', in *IEEE International Conference on Internet of Things, iThings, IEEE International Conference on Green Computing and Communications, and IEEE International Conference on Cyber-Physical-Social Computing*, 2014, pp. 292–295.

[54] M. H. Asghar and N. Mohammadzadeh, 'Design and simulation of energy efficiency in node based on MQTT protocol in Internet of Things', in *International Conference on Green Computing and Internet of Things, ICGCIoT*, 2016, pp. 1413–1417.

[55] H. Hada and J. Mitsugi, 'EPC based internet of things architecture', *IEEE International Conference on RFID-Technologies and Applications, RFID-TA*, pp. 527–532, 2011.

[56] V. Lampkin, W. T. Leong, L. Olivera, S. Rawat, N. Subrahmanyam and R. Xiang, *Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry*, 2012. [Online]. Available: https://www.redbooks.ibm.com/redbooks/pdfs/sg248054.pdf.

[57] C. Prehofer, 'Models at REST or modelling RESTful interfaces for the Internet of Things', in *IEEE World Forum on Internet of Things*, 2015, pp. 251–255.

[58] T. Yokotani and Y. Sasaki, 'Transfer protocols of tiny data blocks in IoT and their performance evaluation', in *IEEE 3rd World Forum on Internet of Things*, 2017, pp. 54–57.

[59] I. Florea, R. Rughinis, L. Ruse and D. Dragomir, 'Survey of Standardized Protocols for the Internet of Things', *Proceedings - 2017 21st International Conference on Control Systems and Computer, CSCS 2017*, pp. 190–196, 2017.

[60] P. Pongle and G. Chavan, 'A survey: Attacks on RPL and 6LoWPAN in IoT', *International Conference on Pervasive Computing: Advance Communication Technology and Application for Society, ICPC*, pp. 1–6, 2015.

[61] Alamsyah, E. Setijadi, I. K. E. Purnama and M. H. Pumomo, 'Performance Comparative Study of AODV, AOMDV and DSDV Routing Protocols in MANET Using NS2', in *International Seminar on Application for Technology of Information and Communication (iSemantic)*, 2018, pp. 286–289.

[62] M. Yamada, T. Oda, Y. Liu, M. Hiyama, K. Matsuo and L. Barolli, 'Performance Evaluation of an IoT-based e-Learning Testbed Considering OLSR and WEP Protocols', in *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, 2016, pp. 335–341.

[63] Z. Guo, I. G. Harris, C. B. Harris, Y. Jiang and L. F. Tsaur, 'A residual battery-aware routing algorithm based on DSR for BLE sensor networks', in *Wireless Telecommunications Symposium (WTS)*, vol. 2016-May, 2016, pp. 1–6.

[64] R. Stewart, 'Stream Control Transmission Protocol', *IETF RFC 4960*, 2007. [Online]. Available: https://tools.ietf.org/html/rfc4960.

[65] A. Ford, C. Raiciu, M. Handley, S. Barre and J. Iyengar, 'Architectural Guidelines for Multipath TCP Development', *IETF RFC 6182*, 2011. [Online]. Available: https://tools.ietf.org/html/rfc6182.

[66] E. D. Ayele, C. Hakkenberg, J. P. Meijers, K. Zhang, N. Meratnia and P. J. M. Havinga, 'Performance analysis of LoRa radio for an indoor IoT applications', in *International Conference on Internet of Things for the Global Community (IoTGC)*, 2017, pp. 1–8.

[67] M. Ha, S. H. Kim and D. Kim, 'Intra-MARIO: A fast mobility management protocol for 6LoWPAN', *IEEE Transactions on Mobile Computing*, vol. 16, no. 1, pp. 172–184, 2017.

[68] T. Kim, S. H. Kim, J. Yang, S. E. Yoo and D. Kim, 'Neighbor table based shortcut tree routing in ZigBee wireless networks', *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 706–716, 2014.

[69] G. Sharma, N. Pandey, I. Hussain and S. K. Kathri, 'Design of framework and analysis of Internet of things at data link layer', in *2nd International Conference on Telecommunication and Networks, TEL-NET 2017*, 2017, pp. 1–4.

[70] A. Ghosh, R. Ratasuk, B. Mondal, N. Mangalvedhe and T. Thomas, 'LTE-advanced: Next-generation wireless broadband technology', *IEEE Wireless Communications*, vol. 17, no. 3, pp. 10–22, 2010.

[71] C. W. Tseng, C. M. Chang and C. H. Huang, 'Complex sensing event process of IoT application based on EPCglobal architecture and IEEE 1451', *Proceedings of 2012 International Conference on the Internet of Things, IOT 2012*, pp. 92–98, 2012.

[72] R. Akeela and Y. Elziq, 'Design and verification of IEEE 802.11ah for IoT and M2M applications', *2017 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2017*, pp. 491–496, 2017.

[73] M. Alnuaimi, K. Shuaib and I. Jawhar, 'Performance Evaluation of IEEE 802.15.4 Physical Layer Using MatLab/Simulink', *IEEE 2006 Innovations in Information Technology*, pp. 1–5, 2006.

[74] M. B. Yassein, W. Mardini and A. Khalil, 'Smart homes automation using Z-wave protocol', *Proceedings - 2016 International Conference on Engineering and MIS, ICEMIS 2016*, 2016.

[75] C.-H. Liao, G. Zhu, D. Kuwabara, M. Suzuki and H. Morikawa, 'Multi-hop LoRa Networks Enabled by Concurrent Transmission', *IEEE Access*, vol. 3536, no. c, pp. 1–1, 2017. [Online]. Available: http : / / ieeexplore . ieee . org / document/8048465/.

[76] A. Floris and L. Atzori, 'Quality of Experience in the Multimedia Internet of Things: Definition and practical use-cases', *2015 IEEE International Conference on Communication Workshop, ICCW 2015*, pp. 1747–1752, 2015.

[77] M. Khan, B. N. Silva and K. Han, 'Internet of Things Based Energy Aware Smart Home Control System', *IEEE Access*, vol. 4, pp. 7556–7566, 2016.

[78] *IoT Connectivity Hub: Smart Business Control*, 2018. [Online]. Available: https://iot.telefonica.com/iot-connectivity-hub/smart-business-control.

[79] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran and S. Guizani, 'Internet-of-Things-Based Smart Cities: Recent Advances and Challenges', *IEEE Communications Magazine*, vol. 55, no. 9, pp. 16–24, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/8030479/.

[80] T. Dimitriou and G. Karame, 'Enabling Anonymous Authorization and Rewarding in the Smart Grid', *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 5, pp. 565–572, 2015.

[81] A. Ukil, S. Bandyoapdhyay, C. Puri and A. Pal, 'IoT healthcare analytics: The importance of anomaly detection', *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, vol. 2016-May, pp. 994–997, 2016.

[82] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal and B. Sikdar, 'A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures', *IEEE Access*, vol. 7, pp. 82 721–82 743, 2019.

[83] S. Raza, S. Duquennoy, J. Höglund, U. Roedig and T. Voigt, 'Secure communication for the Internet of Things – a comparison of link-layer security and IPsec for 6LoWPAN', *Security Comm. Networks*, vol. 7, no. 12, pp. 2654–2668, 2014.

[84] G. Glissa and A. Meddeb, '6LoWPAN multi-layered security protocol based on IEEE 802.15.4 security features', *2017 13th International Wireless Communications and Mobile Computing Conference, IWCMC 2017*, pp. 264–269, 2017.

[85] J. Granjal, E. Monteiro and J. Sa Silva, 'Security for the internet of things: A survey of existing protocols and open research issues', *IEEE Communications Surveys and Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[86] M. S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli and M. H. Rehmani, 'Applications of Blockchains in the Internet of Things: A Comprehensive Survey', *IEEE Communications Surveys and Tutorials*, vol. 21, no. 2, pp. 1676–1717, 2019.

[87] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila and T. Taleb, 'Survey on multi-access edge computing for internet of things realization', *IEEE Communications Surveys Tutorials*, pp. 1–1, 2018.

[88] O. Ormond, G.-M. Muntean and J. Murphy, 'Network Selection Strategy in Heterogeneous Wireless Networks', *Information Technology and Telecommunications Conference (ITT)*, Oct. 2005.

[89] A. Hava, Y. Ghamri-Doudane, G.-M. Muntean and J. Murphy, 'Increasing user perceived quality by selective load balancing of video traffic in wireless networks', *IEEE Transactions on Broadcasting*, vol. 61, no. 2, pp. 238–250, Jun. 2015.

[90] C. H. Muntean and J. McManis, 'A qos-aware adaptive web-based system', in *IEEE International Conference on Communications*, vol. 4, Jun. 2004, pp. 2204–2208.

[91] G.-M. Muntean, 'Efficient delivery of multimedia streams over broadband networks using qoas', *IEEE Transactions on Broadcasting*, vol. 52, no. 2, pp. 230–235, Jun. 2006.

[92] *Cisco Visual Networking Index: Forecast and Trend*, 2017-2022. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.pdf.

[93] A. Castellani and M. Dissegna, 'WebIoT: A web application framework for the internet of things', *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 202–207, 2012.

[94] V. Kepuska and G. Bohouta, 'Next-Generation of Virtual Personal Assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)', in *Proc. of the IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018, pp. 99–103.

[95] Q. Wang, Y. Zhao, W. Wang, D. Minoli, K. Sohraby, H. Zhu and B. Occhiogrosso, 'Multimedia IoT Systems and Applications', in *IEEE Global Internet of Things Summit (GIoTS)*, 2017, pp. 1–6.

[96] N. Singh and S. Singh, 'Virtual Reality: A Brief Survey', in *Proc. of the International Conference on Information Communication and Embedded Systems (ICICES)*, Feb. 2017, pp. 1–6.

[97] Z. Zhang, M. Zhang, Y. Chang, E.-S. Aziz, S. K. Esche and C. Chassapis, 'Collaborative Virtual Laboratory Environments with Hardware in the Loop', in *Cyber-Physical Laboratories in Engineering and Science Education*, Cham: Springer International Publishing, 2018, pp. 363–402.

[98] M. Suznjevic, M. Mandurov and M. Matijasevic, 'Performance and QoE assessment of HTC Vive and Oculus Rift for pick-and-place tasks in VR', *2017 9th International Conference on Quality of Multimedia Experience, QoMEX 2017*, pp. 1–3, 2017.

[99] P. Lelyveld, 'Virtual Reality Primer with an Emphasis on Camera-Captured VR', *SMPTE Motion Imaging Journal*, vol. 124, no. 6, pp. 78–85, Sep. 2015.

[100] M. Narbutt, S. O'Leary, A. Allen, J. Skoglund and A. Hines, 'Streaming VR for immersion: Quality aspects of compressed spatial audio', in *Proceedings of the 2017 23rd International Conference on Virtual Systems and Multimedia, VSMM*, 2017, pp. 1–6.

[101] H. G. Kim, H. Lim and Y. M. Ro, 'Deep Virtual Reality Image Quality Assessment with Human Perception Guider for Omnidirectional Image', *IEEE Trans. Circuits Syst. Video Technol.*, vol. PP, pp. 1–11, 2019.

[102] G. Regal, R. Schatz, J. Schrammel and S. Suette, 'VRate: A Unity3D Asset for integrating Subjective Assessment Questionnaires in Virtual Environments', *2018 10th International Conference on Quality of Multimedia Experience, QoMEX 2018*, pp. 1–3, 2018.

[103] M. Hubbell and J. Kepner, 'Large scale network situational awareness via 3D gaming technology', *2012 IEEE Conference on High Performance Extreme Computing (HPEC)*, pp. 1–5, 2012.

[104] I. Toumpalidis, K. Cheliotis, F. Roumpani and A. Hudson-Smith, 'Vr binoculars: An immersive visualization framework for iot data streams', in *Living in the Internet of Things: Cybersecurity of the IoT - 2018*, Mar. 2018, pp. 1–7.

[105] J. Westlin and T. H. Laine, 'Short paper: Calory battle ar: An extensible mobile augmented reality exergame platform', in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, Mar. 2014, pp. 171–172.

[106] A. E. Staiano and S. L. Calvert, 'Exergames for Physical Education Courses: Physical, Social, and Cognitive Benefits', en, *Child Development Perspectives*, vol. 5, no. 2, pp. 93–98, Jun. 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1750-8606.2011.00162.x.

[107] M. Alessi, E. Giangreco, M. Pinnella, S. Pino, D. Storelli, L. Mainetti, V. Mighali and L. Patrono, 'A Web Based Virtual Environment as a Connection Platform between People and IoT', in *Proc. of the 2016 International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*, 2016, pp. 1–6.

[108] L. Atzori, A. Iera, G. Morabito and M. Nitti, 'The social internet of things (SIoT) - When social networks meet the internet of things: Concept, architecture and network characterization', *Computer Networks*, vol. 56, no. 16, pp. 3594–3608, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2012.07.010.

[109] P. Cooper, *28 Twitter Statistics All Marketers Should Know in 2019*, 2019. [Online]. Available: https://blog.hootsuite.com/twitter-statistics/.

[110] N. O'Leary, *GitHub - Twitter to MQTT Publisher*, 2012. [Online]. Available: https://github.com/knolleary/twitter-to-mqtt.

[111] J.-P. Mens, *GitHub - MQTWIT*, 2014. [Online]. Available: https://github.com/jpmens/mqtwit.

[112] D. Sheridan, A. A. Simiscuka and G.-M. Muntean, 'Design, Implementation and Analysis of a Twitter-based Social IoT Network', in *International Symposium on Sensors and Instrumentation in IoT Era*, Lisbon, 2019.

[113] J.-P. Mens, *GitHub - mqttwarn*, 2019. [Online]. Available: https://github.com/jpmens/mqttwarn.

[114] J. Höller, V. Tsiatsis, C. Mulligan, S. Karnouskos, S. Avesand and D. Boyle, 'M2M to IoT – An Architectural Overview', in *From Machine-To-Machine to the Internet of Things: Introduction to a New Age of Intelligence*, Elsevier, 2014, pp. 61–77.

[115] C. Bormann, J. Jimenez and A. Melnikov, *Constrained RESTful Environments (core)*, 2018. [Online]. Available: https://datatracker.ietf.org/wg/core/charter/.

[116] T. Leppänen, J. Riekki, M. Liu, E. Harjula and T. Ojala, 'Mobile Agents-Based Smart Objects for the Internet of Things', in *Internet of Things Based on Smart Objects*, Springer, Cham, 2014, pp. 29–48.

[117] A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, 'Internet of Things for Smart Cities', *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[118] D. Guinard, V. Trifa and E. Wilde, 'A resource oriented architecture for the Web of Things', *IEEE International Conference on the Internet of Things*, pp. 1–8, 2010.

[119] C. Mulligan and M. Olsson, 'Architectural Implications of Smart City Business Models: An Evolutionary Perspective', *IEEE Communications Magazine*, vol. 51, no. 6, pp. 80–85, 2013.

[120] Zhihong Yang, Yufeng Peng, Yingzhao Yue, Xiaobo Wang, Yu Yang and Wenji Liu, 'Study and application on the architecture and key technologies for IOT', *International Conference on Multimedia Technology*, pp. 747–751, 2011.

[121] R. Khan, S. U. Khan, R. Zaheer and S. Khan, 'Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges', in *International Conference on Frontiers of Information Technology (FIT)*, 2012, pp. 257–260.

[122] S. A. Alvi, B. Afzal, G. A. Shah, L. Atzori and W. Mahmood, 'Internet of multimedia things: Vision and challenges', *Ad Hoc Networks*, vol. 33, pp. 87–111, Oct. 2015.

[123] A. Hazmi, J. Rinne and M. Valkama, 'Feasibility study of IEEE 802.11ah radio technology for IoT and M2M use cases', *IEEE Globecom Workshops*, pp. 1687–1692, 2012.

[124] T. A. Butt, I. Phillips, L. Guan and G. Oikonomou, 'Adaptive and context-aware service discovery for the Internet of Things', in *13th International Conference on Next Generation Wired/Wireless Advanced Networking & 6th Conference on Internet of Things and Smart Spaces (NEW2AN/ruSMART)*, vol. 8121, 2013, pp. 36–47.

[125] S. Andreev, M. Gerasimenko, O. Galinina, Y. Koucheryavy, N. Himayat, S. P. Yeh and S. Talwar, 'Intelligent access network selection in converged multi-radio heterogeneous networks', *IEEE Wireless Communications*, vol. 21, no. 6, pp. 86–96, 2014.

[126] L. Zou, R. Trestian and G. M. Muntean, 'EDOAS: Energy-aware device-oriented adaptive multimedia scheme for Wi-Fi offload', in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2014, pp. 2916–2921.

[127] S. Ezdiani, I. S. Acharyya, S. Sivakumar and A. Al-Anbuky, 'An IoT Environment for WSN Adaptive QoS', in *IEEE International Conference on Data Science and Data Intensive Systems*, 2015, pp. 586–593.

[128] L. Li, S. Li and S. Zhao, 'QoS-Aware scheduling of services-oriented internet of things', *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1497–1507, 2014.

[129] T. Petrić, M. Goessens, L. Nuaymi, L. Toutain and A. Pelov, 'Measurements, performance and analysis of LoRa FABIAN, a real-world implementation of LPWAN', in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, 2016, pp. 1–7.

[130] R. Hassan, A. M. Jubair, K. Azmi and A. Bakar, 'Adaptive Congestion Control Mechanism in CoAP Application Protocol For Internet of Things (IoT)', in *International Conference on Signal Processing and Communication (ICSC)*, 2016, pp. 121–125.

[131] M. L. M. Peixoto, D. L. Filho, C. Henrique, D. Segura, B. Tardiole and B. Guazzelli, 'Predictive dynamic algorithm: An approach toward QoS-aware service for IoT-cloud environment', in *Proc. 16th IEEE International Conference on Computer and Information Technology, CIT 2016, 2016 6th International Symposium on Cloud and Service Computing*, 2017, pp. 686–693.

[132] I. Awan, M. Younas and W. Naveed, 'Modelling QoS in IoT applications', in *Proc. 2014 International Conference on Network-Based Information Systems, NBiS 2014*, 2014, pp. 99–105.

[133] B. K. J. Al-Shammari, N. A. Al-Aboody and H. S. Al-Raweshidy, 'IoT Traffic Management and Integration in the QoS Supported Network', *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 352–370, 2017.

[134] A. Dvornikov, P. Abramov, S. Efremov and L. Voskov, 'QoS Metrics Measurement in Long Range IoT Networks', in *IEEE 19th Conference on Business Informatics (CBI)*, 2017, pp. 15–20.

[135] S. Y. Yu, C. S. Shih, J. Y. J. Hsu, Z. Huang and K. J. Lin, 'QoS oriented sensor selection in IoT system', in *IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)*, 2014, pp. 201–206.

[136] H. Elhammouti, E. Sabir, M. Benjillali, L. Echabbi and H. Tembine, 'Self-Organized Connected Objects: Rethinking QoS Provisioning for IoT Services', *IEEE Communications Magazine*, vol. 55, no. 9, pp. 41–47, 2017.

[137] M. E. Khanouche, Y. Amirat, A. Chibani, M. Kerkar and A. Yachir, 'Energy-Centered and QoS-Aware Services Selection for Internet of Things', *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1256–1269, 2016.

[138] R. Zhang, M. Wang, X. Shen and L. L. Xie, 'Probabilistic Analysis on QoS Provisioning for Internet of Things in LTE-A Heterogeneous Networks with Partial Spectrum Usage', *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 354–365, 2016.

[139] A. Botta, A. Pescapé and G. Ventre, 'Quality of service statistics over heterogeneous networks: Analysis and applications', *European Journal of Operational Research*, vol. 191, no. 3, pp. 1075–1088, 2008.

[140] L. Zhou, L. Chen, H. Pung and L. Ngoh, 'Identifying QoS violations through statistical end-to-end analysis', *International Journal of Communication Systems*, no. 24, pp. 1388–1406, 2011.

[141] G. Horvat, D. Žagar and D. Vinko, 'Influence of Node Deployment Parameters on QoS in Large-Scale WSN', in *IEEE Mediterranean Conference on Embedded Computing (MECO)*, 2014, pp. 202–205.

[142] A. Valehi and A. Razi, 'Maximizing Energy Efficiency of Cognitive Wireless Sensor Networks With Constrained Age of Information', *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 643–654, 2017.

[143] S. Nath, J. Wu and J. Yang, 'Optimizing Age-of-Information and Energy Efficiency Tradeoff for Mobile Pushing Notifications', in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017, pp. 380–384.

[144] A. Baiocchi and I. Turcanu, 'A Model for the Optimization of Beacon Message Age-of-Information in a VANET', in *International Teletraffic Congress (ITC)*, 2017, pp. 108–116.

[145] Q. He, D. Yuan and A. Ephremides, 'Optimizing freshness of information: On minimum age link scheduling in wireless systems', in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2016.

[146] Q. He, D. Yuan and A. Ephremides, 'Optimal Link Scheduling for Age Minimization in Wireless Systems', *IEEE Transactions on Information Theory*, vol. 64, no. 7, pp. 5381–5394, 2018.

[147] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal and N. B. Shroff, 'Update or Wait: How to Keep Your Data Fresh', *IEEE Transactions on Information Theory*, vol. 63, no. 11, pp. 7492–7508, 2017.

[148] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier and A. Ephremides, 'Modeling the Age of Information in Emulated Ad Hoc Networks', in *IEEE Military Communications Conference (MILCOM)*, 2017, pp. 436–441.

[149] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier and A. Ephremides, 'Information freshness and popularity in mobile caching', in *IEEE International Symposium on Information Theory*, 2017, pp. 136–140.

[150] O. Bello and S. Zeadally, 'Intelligent Device-to-Device Communication in the Internet of Things', *IEEE Systems Journal*, vol. 10, no. 3, pp. 1172–1182, 2016.

[151] B.-L. Wenning, A. Timm-Giel and C. Görg, 'A generic framework for context-aware routing and its implementation in wireless sensor networks', in *Technologien und Anwendungen. Vorträge der 14. ITG-Fachtagung*, 2009, pp. 53–58.

[152] A. Aksoy and M. H. Gunes, 'Automated IoT Device Identification using Network Traffic', in *IEEE International Conference on Communications (ICC)*, IEEE, 2019, pp. 1–7.

[153] R. Morabito, I. Farris, A. Iera and T. Taleb, 'Evaluating Performance of Containerized IoT Services for Clustered Devices at the Network Edge', *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 1019–1030, 2017.

[154] S. Li, G. Oikonomou, T. Tryfonas, T. M. Chen and L. D. Xu, 'A distributed consensus algorithm for decision making in service-oriented internet of things', *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1461–1468, 2014.

[155] N. Nasser, L. Karim, A. Ali and N. Khelifi, 'Multiple Base station and Packet Priority-based clustering scheme in Internet of Things', in *International Conference on Computing, Management and Telecommunications (ComManTel)*, 2014, pp. 58–61.

[156] W. Twayej, M. Khan and H. S. Al-Raweshidy, 'Network Performance Evaluation of M2M with Self Organizing Cluster Head to Sink Mapping', *IEEE Sensors Journal*, vol. 17, no. 15, pp. 4962–4974, 2017.

[157] T. Al-Janabi and H. Al-Raweshidy, 'Optimised Clustering Algorithm-Based Centralised Architecture for Load Balancing in IoT Network', in *International Symposium on Wireless Communication Systems (ISWCS)*, 2017, pp. 269–274.

[158] S. Balasubramaniam and R. Jagannath, 'A Service Oriented IoT Using Cluster Controlled Decision Making', in *IEEE International Advance Computing Conference (IACC)*, 2015, pp. 558–563.

[159] S. Ghosh, 'Mechanism for adaptive and context-aware inter- IoT communication', in *IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)*, 2015, pp. 1–6.

[160] J. Kim and J. Lee, 'Cluster-Based Mobility Supporting WMN for IoT Networks', in *IEEE International Conference on Green Computing and Communications (GreenCom)*, 2012, pp. 700–703.

[161] L. Song, K. K. Chai, Y. Chen, J. Schormans, J. Loo and A. Vinel, 'QoS-Aware Energy-Efficient Cooperative Scheme for Cluster-Based IoT Systems', *IEEE Systems Journal*, vol. 11, no. 3, pp. 1447–1455, 2017.

[162] S. A. Nikolidakis, D. Kandris, D. D. Vergados and C. Douligeris, 'Energy efficient routing in wireless sensor networks through balanced clustering', *Algorithms*, vol. 6, no. 1, pp. 29–42, 2013.

[163] D. Puschmann, P. Barnaghi and R. Tafazolli, 'Adaptive Clustering for Dynamic IoT Data Streams', *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 64–74, 2017.

[164] H. Rahimi, A. Zibaeenejad and A. A. Safavi, 'A Novel IoT Architecture based on 5G-IoT and Next Generation Technologies', *IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pp. 81–88, 2018.

[165] Z. Zhou, M. Dong, K. Ota, G. Wang and L. T. Yang, 'Energy-Efficient Resource Allocation for D2D Communications Underlaying Cloud-RAN-Based LTE-A Networks', *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 428–438, 2016.

[166] X. Liu and N. Ansari, 'Green Relay Assisted D2D Communications with Dual Batteries in Heterogeneous Cellular Networks for IoT', *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1707–1715, 2017.

[167] L. W. Chen and J. J. Chung, 'Mobility-Aware and Congestion-Relieved Dedicated Path Planning for Group-Based Emergency Guiding Based on Internet of Things Technologies', *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2453–2466, 2017.

[168] F. Al-Turjman, 'Energy-Aware Data Delivery Framework for Safety-Oriented Mobile IoT', *IEEE Sensors Journal*, vol. 18, no. 1, pp. 470–478, 2017.

[169] C. M. Huang, C. H. Shao, S. Z. Xu and H. Zhou, 'The Social Internet of Thing (S-IOT)-Based Mobile Group Handoff Architecture and Schemes for Proximity Service', *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 3, pp. 425–437, 2017.

[170] M. Ishino, Y. Koizumi and T. Hasegawa, 'A study on a routing-based mobility management architecture for IoT devices', in *International Conference on Network Protocols (ICNP)*, 2014, pp. 498–500.

[171] A. Jara, M. Zamora and A. Skarmeta, 'An architecture based on internet of things to support mobility and security in medical environments', *IEEE Consumer Communications and Networking Conference (CCNC)*, 2010.

[172] C. Perera, P. P. Jayaraman, A. Zaslavsky, D. Georgakopoulos and P. Christen, 'MOSDEN: An internet of things middleware for resource constrained mobile devices', in *Annual Hawaii International Conference on System Sciences*, 2014, pp. 1053–1062.

[173] J. Han, J. Yun, J. Jang and K. R. Park, 'User-Friendly Home Automation Based on 3D Virtual World', *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1843–1847, Sep. 2010.

[174] Y. Jeong, H. Joo, G. Hong, D. Shin and S. Lee, 'AVIoT: Web-Based Interactive Authoring and Visualization of Indoor Internet of Things', *IEEE Trans. Consum. Electron.*, vol. 61, no. 3, pp. 295–301, Sep. 2015.

[175] L. Czekierda, S. Zielinski and M. Szreter, 'Benefits of Extending Collaborative Educational Cloud with IoT', in *IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2017, pp. 80–85.

[176] J. Huang, Q. Duan, Y. Zhao, Z. Zheng and W. Wang, 'Multicast Routing for Multimedia Communications in the Internet of Things', *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 215–224, Feb. 2017.

[177] M. Di Francesco, N. Li, M. Raj and S. K. Das, 'A storage infrastructure for heterogeneous and multimedia data in the Internet of Things', in *IEEE International Conference on Green Computing and Communications*, 2012, pp. 26–33.

[178] C. G. Coogan and B. He, 'Brain-Computer Interface Control in a Virtual Reality Environment and Applications for the Internet of Things', *IEEE Access*, vol. 6, pp. 10 840–10 849, 2018.

[179] D. You, B. S. Seo, E. Jeong and D. H. Kim, 'Internet of Things (IoT) for seamless virtual reality space: Challenges and perspectives', *IEEE Access*, vol. 6, pp. 40 439–40 449, 2018.

[180] Z. Lv, T. Yin, H. Song and G. Chen, 'Virtual Reality Smart City Based on WebVRGIS', *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1015–1024, Dec. 2016.

[181] M. I. Choi, L. W. Park, S. Lee, J. Y. Hwang and S. Park, 'Design and Implementation of Hyper-connected IoT-VR Platform for Customizable and Intuitive Remote Services', in *IEEE International Conference on Consumer Electronics (ICCE)*, 2017, pp. 1–2.

[182] F. Fittkau, A. Krause and W. Hasselbring, 'Exploring Software Cities in Virtual Reality', in *Proc. of the IEEE 3rd Working Conference on Software Visualization*, 2015, pp. 130–134.

[183] T. T. Le, D. V. Nguyen and E. S. Ryu, 'Computing Offloading over mmWave for Mobile VR: Make 360 Video Streaming Alive', *IEEE Access*, vol. 6, pp. 66 576–66 589, 2018.

[184] A. R. Biswas and R. Giaffreda, 'IoT and Cloud Convergence: Opportunities and Challenges', in *Proc. of the 2014 IEEE World Forum on Internet of Things (WF-IoT)*, Mar. 2014, pp. 375–376.

[185] Y.-T. Lee, W.-H. Hsiao, C.-M. Huang and S.-C. Chou, 'An Integrated Cloud-Based Smart Home Management System with Community Hierarchy', *IEEE Trans. Consum. Electron.*, vol. 62, no. 1, pp. 1–9, Feb. 2016.

[186] Y. Benazzouz, C. Munilla, O. Gunalp, M. Gallissot and L. Gurgen, 'Sharing User IoT Devices in the Cloud', in *Proc.of the 2014 IEEE World Forum on Internet of Things (WF-IoT)*, 2014, pp. 373–374.

[187] J. Delsing, J. Eliasson, J. Deventer, H. Derhamy and P. Varga, 'Enabling IoT Automation Using Local Clouds', in *Proc. of the IEEE World Forum on Internet of Things (WF-IoT)*, 2016, pp. 502–507.

[188] X. Hou, Y. Lu and S. Dey, 'Wireless VR/AR with Edge/Cloud Computing', in *Proc. of the 26th International Conference on Computer Communication and Networks (ICCCN)*, 2017, pp. 1–8.

[189] Y. Kang, H. Kim and J. Kang, 'Docker Based Computation Off-Loading for Video Game Based Mobile VR Framework', in *Proc. of the 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2017, pp. 123–125.

[190] M. A. Alharbe, 'Awarenessability and influences on raising of traffic accidents through the content of social media in the internet of things: A practical empirical study by the internet of things and multimedia on university students in western Saudi Arabia', in *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2018*, 2018, pp. 48–51.

[191] P. Yenkar and S. D. Sawarkar, 'A survey on social media analytics for smart city', in *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2018*, 2019, pp. 87–93.

[192] B. Jadhav and S. C. Patil, 'Wireless Home monitoring using Social Internet of Things (SIoT)', in *International Conference on Automatic Control and Dynamic Optimization Techniques, ICACDOT 2016*, 2016, pp. 925–929.

[193] Waze, *Waze*, 2019. [Online]. Available: https://www.waze.com/en-GB/.

[194] Qualcomm, *Qualcomm Announces the Launch of HealthyCircles Mobile*, 2015. [Online]. Available: https://www.qualcomm.com/news/releases/2015/04/08.

[195] C. Allison, *And finally: Nike to debut connected shoes, but don't expect them to track your run*, 2018. [Online]. Available: https://www.wareable.com/wearable-tech/nike-connected-shoe-2801.

[196] C. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos and E. Kayafas, 'License plate recognition from still images and video sequences: A survey', *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 377–391, Sep. 2008.

[197] S. He, Y. Yuan, C. Fu, X. Hu and Y. Zhao, 'Robust license plate detection using profile-based filter', in *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, Mar. 2018, pp. 794–800.

[198] A. M. Al-Ghaili, S. Mashohor, A. R. Ramli and A. Ismail, 'Vertical-edge-based car-license-plate detection method', *IEEE Transactions on Vehicular Technology*, vol. 62, no. 1, pp. 26–38, Jan. 2013.

[199] J.-W. Hsieh, S.-H. Yu and Y.-S. Chen, 'Morphology-based license plate detection from complex scenes', in *Object recognition supported by user interaction for service robots*, vol. 3, Aug. 2002, 176–179 vol.3.

[200] A. A. Lensky, K. Jo and V. V. Gubarev, 'Vehicle license plate detection using local fractal dimension and morphological analysis', in *2006 International Forum on Strategic Technology*, Oct. 2006, pp. 47–50.

[201] L. Luo, H. Sun, W. Zhou and L. Luo, 'An efficient method of license plate location', in *2009 First International Conference on Information Science and Engineering*, Dec. 2009, pp. 770–773.

[202] L. Hu and Q. Ni, 'Iot-driven automated object detection algorithm for urban surveillance systems in smart cities', *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 747–754, Apr. 2018.

[203] O. Bulan, V. Kozitsky, P. Ramesh and M. Shreve, 'Segmentation- and annotation-free license plate recognition with deep localization and failure identification', *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2351–2363, Sep. 2017.

[204] M. A. Rafique, W. Pedrycz and M. Jeon, 'Vehicle license plate detection using region-based convolutional neural networks', en, *Soft Computing*, pp. 1–12, Jun. 2017.

[205] T. Juhana and V. G. Anggraini, 'Design and implementation of smart home surveillance system', in *2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA)*, Oct. 2016, pp. 1–5.

[206] L. Chen, C.-R. Chen and D.-E. Chen, 'Vips: A video-based indoor positioning system with centimeter-grade accuracy for the iot', in *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Mar. 2017, pp. 63–65.

[207] M. Alam, J. Ferreira, S. Mumtaz, M. A. Jan, R. Rebelo and J. A. Fonseca, 'Smart cameras are making our beaches safer: A 5g-envisioned distributed architecture for safe, connected coastal areas', *IEEE Vehicular Technology Magazine*, vol. 12, no. 4, pp. 50–59, Dec. 2017.

[208] A. Giyenko and Y. I. Cho, 'Intelligent uav in smart cities using iot', in *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, Oct. 2016, pp. 207–210.

[209] H. G. Seif and X. Hu, 'Autonomous Driving in the iCity—HD Maps as a Key Challenge of the Automotive Industry', *Engineering*, vol. 2, no. 2, pp. 159–162, 2016.

[210] J. Jiao, 'Machine learning assisted high-definition map creation', in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 01, Jul. 2018, pp. 367–373.

[211] J. J. P. C. Rodrigues, D. B. D. R. Segundo, H. A. Junqueira, M. H. Sabino, R. M. Prince, J. Al-Muhtadi and V. H. C. D. Albuquerque, 'Enabling technologies for the internet of health things', *IEEE Access*, vol. 6, pp. 13 129–13 141, 2018.

[212] C. Doukas and I. Maglogiannis, 'Bringing iot and cloud computing towards pervasive healthcare', in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, Jul. 2012, pp. 922–926.

[213] A. Dohr, R. Modre-Opsrian, M. Drobics, D. Hayn and G. Schreier, 'The internet of things for ambient assisted living', in *Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations*, ser. ITNG '10, Washington, DC, USA: IEEE Computer Society, 2010, pp. 804–809.

[214] M. M. Iqbal, M. Farhan, S. Jabbar, Y. Saleem and S. Khalid, 'Multimedia based IoT-centric smart framework for eLearning paradigm', en, *Multimedia Tools and Applications*, pp. 1–20, Feb. 2018.

[215] S. Creane, Y. Crotty and M. Farren, 'A proposed use of virtual and augmented reality for supporting inquiry based learning', in *2015 International Conference on Interactive Mobile Communication Technologies and Learning (IMCL)*, Nov. 2015, pp. 393–395.

[216] M. S. Mekala and P. Viswanathan, 'A novel technology for smart agriculture based on iot with cloud computing', in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Feb. 2017, pp. 75–82.

[217] J.-c. Zhao, J.-f. Zhang, Y. Feng and J.-x. Guo, 'The study and application of the iot technology in agriculture', in *2010 3rd International Conference on Computer Science and Information Technology*, vol. 2, Jul. 2010, pp. 462–465.

[218] R. Nukala, K. Panduru, A. Shields, D. Riordan, P. Doody and J. Walsh, 'Internet of things: A review from 'farm to fork'', in *2016 27th Irish Signals and Systems Conference (ISSC)*, Jun. 2016, pp. 1–6.

[219] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, 'Internet of Things: A Survey on Enabling Technologies, Protocols and Applications', *IEEE Communications Surveys & Tutorials*, vol. PP, no. 99, pp. 1–1, 2015. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7123563.

[220] A. A. Simiscuka, M. Bezbradica and G.-M. Muntean, 'Performance Analysis of the Quality of Service-Aware Networking Scheme for Smart Internet of Things Gateways', *13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1370–1374, 2017.

[221] A. A. Simiscuka, C. H. Muntean and G.-M. Muntean, 'A Networking Scheme for an Internet of Things Integration Platform', *IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 271–276, 2017.

[222] H. Zhao, L. Zhu, H. Jiang and T. Tang, 'Design and performance tests in an integrated TD-LTE based train ground communication system', in *17th IEEE International Conference on Intelligent Transportation Systems (ITSC)*, Nov. 2014, pp. 747–750.

[223] C. Park, J. Liu and P. H. Chou, 'Eco: An ultra-compact low-power wireless sensor node for real-time motion monitoring', in *4th International Symposium on Information Processing in Sensor Networks, (IPSN)*, 2005, pp. 398–403.

[224] M. Uddin, A. Salem, I. Nam and T. Nadeem, 'Wearable sensing framework for human activity monitoring', in *Proceedings of the Workshop on Wearable Systems and Applications (WearSys)*, May 2015, pp. 21–26.

[225] M. Schmitt, J. Redi, P. Cesar and D. Bulterman, '1Mbps is enough: Video quality and individual idiosyncrasies in multiparty HD video-conferencing', in *8th International Conference on Quality of Multimedia Experience (QoMEX)*, Jun. 2016.

[226] IBM, *SPSS Statistics - Overview*, 2019. [Online]. Available: https://www.ibm.com/products/spss-statistics.

[227] J. Tukey, 'Comparing Individual Means in the Analysis of Variance', *Biometrics*, vol. 5, no. 2, pp. 99–114, 1949.

[228] A. A. Simiscuka and G.-M. Muntean, 'A Relay and Mobility Scheme for QoS Improvement in IoT Communications', in *Proc. of the IEEE International Conference on Communications Workshops (ICC Workshops)*, 2018, pp. 1–6.

[229] A. A. Simiscuka and G. M. Muntean, 'Age of Information as a QoS Metric in a Relay-Based IoT Mobility Solution', in *Proc. of the 14th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2018, pp. 868–873.

[230] G. Muntean, P. Perry and L. Murphy, 'Objective and subjective evaluation of qoas video streaming over broadband networks', *IEEE Transactions on Network and Service Management*, vol. 2, no. 1, pp. 19–28, 2005.

[231] C. H. Muntean and J. McManis, 'End-user quality of experience oriented adaptive e-learning system', *Journal of Digital Information, Special Issue on Adaptive Hypermedia*, vol. 7, no. 1, 2006.

[232] S. Ezdiani, I. S. Acharyya, S. Sivakumar and A. Al-Anbuky, 'Wireless Sensor Network Softwarization: Towards WSN Adaptive QoS', *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1517–1527, 2017.

[233] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran and M. Guizani, 'Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges', *IEEE Wireless Communications*, vol. 24, no. 3, pp. 10–16, 2017.

[234] E. Ahmed, I. Yaqoob, A. Gani, M. Imran and M. Guizani, 'Internet-of-things-based smart environments: State of the art, taxonomy, and open research challenges', *IEEE Wireless Communications*, vol. 23, no. 5, pp. 10–16, 2016.

[235] Y. Li, K. Chi, H. Chen, Z. Wang and Y. Zhu, 'Narrowband internet of things systems with opportunistic D2D communication', *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1474–1484, 2018.

[236] Z. Zhou, K. Ota, M. Dong and C. Xu, 'Energy-Efficient Matching for Resource Allocation in D2D Enabled Cellular Networks', *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 5256–5268, 2017.

[237] Z. Zhou, H. Yu, C. Xu, Y. Zhang, S. Mumtaz and J. Rodriguez, 'Dependable Content Distribution in D2D-Based Cooperative Vehicular Networks: A Big Data-Integrated Coalition Game Approach', *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 953–964, 2018.

[238] M. Bouaziz, A. Rachedi and A. Belghith, 'EC-MRPL : An Energy-Efficient and Mobility Support Routing Protocol for Internet of Mobile Things', in *The 14th Annual IEEE Consumer Communications & Networking Conference*, 2017, pp. 19–24.

[239] C.-C. Lin, D.-J. Deng and L.-Y. Lu, 'Many-Objective Sensor Selection in IoT Systems', *IEEE Wireless Communications*, vol. 24, no. 3, pp. 40–47, 2017.

[240] A. Brogi and S. Forti, 'QoS-aware deployment of IoT applications through the fog', *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1185–1192, 2017.

[241] Q. Fan and N. Ansari, 'Application Aware Workload Allocation for Edge Computing-Based IoT', *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2146–2153, 2018.

[242] F. Bian, D. Kempe and R. Govindan, 'Utility-based Sensor Selection', in *Proc. 5th Int'l. Conf. Info. Processing Sensor Networks*, ACM Press, 2006, pp. 11–18.

[243] S. Kaul, R. Yates and M. Gruteser, 'Real-time status: How often should one update?', in *IEEE INFOCOM*, 2012, pp. 2731–2735.

[244] A. S. Azini, M. R. Kamarudin and M. Jusoh, 'Transparent antenna for WiFi application: RSSI and throughput performances at ISM 2.4 GHz', *Telecommunication Systems*, vol. 61, no. 3, pp. 569–577, Mar. 2016. [Online]. Available: http://link.springer.com/10.1007/s11235-015-0013-x.

[245] O. Said, Y. Albagory, M. Nofal and F. Al Raddady, 'IoT-RTP and IoT-RTCP: Adaptive protocols for multimedia transmission over internet of things environments', *IEEE Access*, vol. 5, pp. 16 757–16 773, 2017.

[246] A. A. Simiscuka and G.-M. Muntean, 'Synchronisation between Real and Virtual-World Devices in a VR-IoT Environment', in *Proc. of the IEEE International Symposium on Broadband Multimedia Systems*, 2018, pp. 1–6.

[247] A. A. Simiscuka, T. M. Markande and G.-M. Muntean, 'Real-Virtual World Device Synchronization in a Cloud-enabled Social Virtual Reality IoT Network', *IEEE Access*, vol. 7, pp. 106 588–106 599, 2019.

[248] E. Rubio-Drosdov, D. Díaz-Sánchez, F. Almenárez, P. Arias-Cabarcos and A. Marín, 'Seamless Human-Device Interaction in the Internet of Things', *IEEE Transactions on Consumer Electronics*, vol. 63, no. 4, pp. 490–498, Nov. 2017.

[249] G.-M. Muntean, P. Perry and L. Murphy, 'Objective and subjective evaluation of QOAS video streaming over broadband networks', *IEEE Transactions on Network and Service Management*, vol. 2, no. 1, pp. 19–28, Nov. 2005.

[250] A. N. Moldovan, A. Molnar and C. H. Muntean, 'EcoLearn: Battery Power Friendly E-Learning Environment for Mobile Device Users', in *Learning-Oriented Technologies, Devices And Networks*, Lambert Academic Publishing, 2011, pp. 273–296.

[251] G.-M. Muntean, P. Perry and L. Murphy, 'Subjective Assessment of the Quality-Oriented Adaptive Scheme', *IEEE Transactions on Broadcasting*, vol. 51, no. 3, pp. 276–286, Aug. 2005.

[252] F. Buttussi and L. Chittaro, 'Effects of Different Types of Virtual Reality Display on Presence and Learning in a Safety Training Scenario', *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 2, pp. 1063–1076, 2018.

[253] A. Steed, S. Friston, M. M. Lopez, J. Drummond, Y. Pan and D. Swapp, 'An 'In the Wild' Experiment on Presence and Embodiment using Consumer Virtual Reality Equipment', *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 4, pp. 1406–1414, 2016.

[254] P. Corcoran, 'A Matter of Timing: Consumer Electronics and Network Time', *IEEE Cons. Electronics Magazine*, vol. 2, no. 4, pp. 20–25, 2013.

[255] *Adafruit IO*, 2019. [Online]. Available: https://io.adafruit.com/.

[256] *Oculus Rift*, 2019. [Online]. Available: https://www.oculus.com/rift/.

[257] *Raspberry Pi*, 2019. [Online]. Available: https://www.raspberrypi.org/.

[258] *Beeks Beacons*, 2017. [Online]. Available: https://adiglobal.cz/iiWWW/docs.nsf/all/D68D4C7E25A040A1C125829B0037E4F5/$FILE/BEEKs_Industrial%20-%20Datasheet_EN.pdf.

[259] *Alienware Aurora*, 2019. [Online]. Available: https://www.dell.com/en-ie/shop/desktops-and-all-in-ones/alienware-aurora/spd/alienware-aurora-r7-desktop.

[260] *GlassFish*, 2019. [Online]. Available: https://javaee.github.io/glassfish/.

[261] *JavaServer Faces*, 2019. [Online]. Available: https://javaee.github.io/javaserverfaces-spec/.

[262] *A-Frame*, 2019. [Online]. Available: https://aframe.io/.

[263] *Twython - Twython 3.6.0 documentation*, 2019. [Online]. Available: https://twython.readthedocs.io/en/latest/.

[264] *Adafruit_IO_Python on GitHub*, 2019. [Online]. Available: https://github.com/adafruit/Adafruit%5C_IO%5C_Python/tree/master/Adafruit%5C_IO.

[265] G. Kobayashi, M. C. Broens, M. E. Q. Gonzalez and J. A. Quilici-Gonzalez, 'The Internet of Things and its Impact on Social Relationships Involving Mutual Trust', in *Proc. of the 2015 IEEE International Symposium on Technology and Society (ISTAS)*, 2015, pp. 1–6.

[266] M. E. Koltko-Rivera, 'The Potential Societal Impact of Virtual Reality', *Advances in Virtual Environments Technology: Musings on Design, Evaluation, and Applications*, vol. 9, pp. 1–18, 2005.