

People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

KASDI MERBAH OUARGLA UNIVERSITY

Department of computer science and information technologies



Master Memoire In Informatique

Speciality: Network Administration and security

Presented by: Tati Amani, BenSaci Cheyma

THEME:

Simulation and evaluation of an IOT communication protocol

Framed by: M.Khelili Khalida Farida

Before the jury:

Mr. Cheradid A KASDI MERBAH UNVERSITY OUARGLA President

Mr. Mahdjoub MB KASDI MERBAH UNVERSITY OUARGLA Examiner

University year 2022-2021



Thanks:

Praise and thanks be to God Almighty, who enabled me to complete my studies with this work, despite its humility, but it is a great achievement for me and my parents. I offer all my thanks to my father, my friends, my teachers, the Department of Information and Communications, all the management staff, and everyone who helped me and worked with me, even if a little. I also dedicate a lot of thanks to Professor Khelili F, who worked hard with us all the time, and to all the professors of the jury, and we ask God to grant you and me success in life.

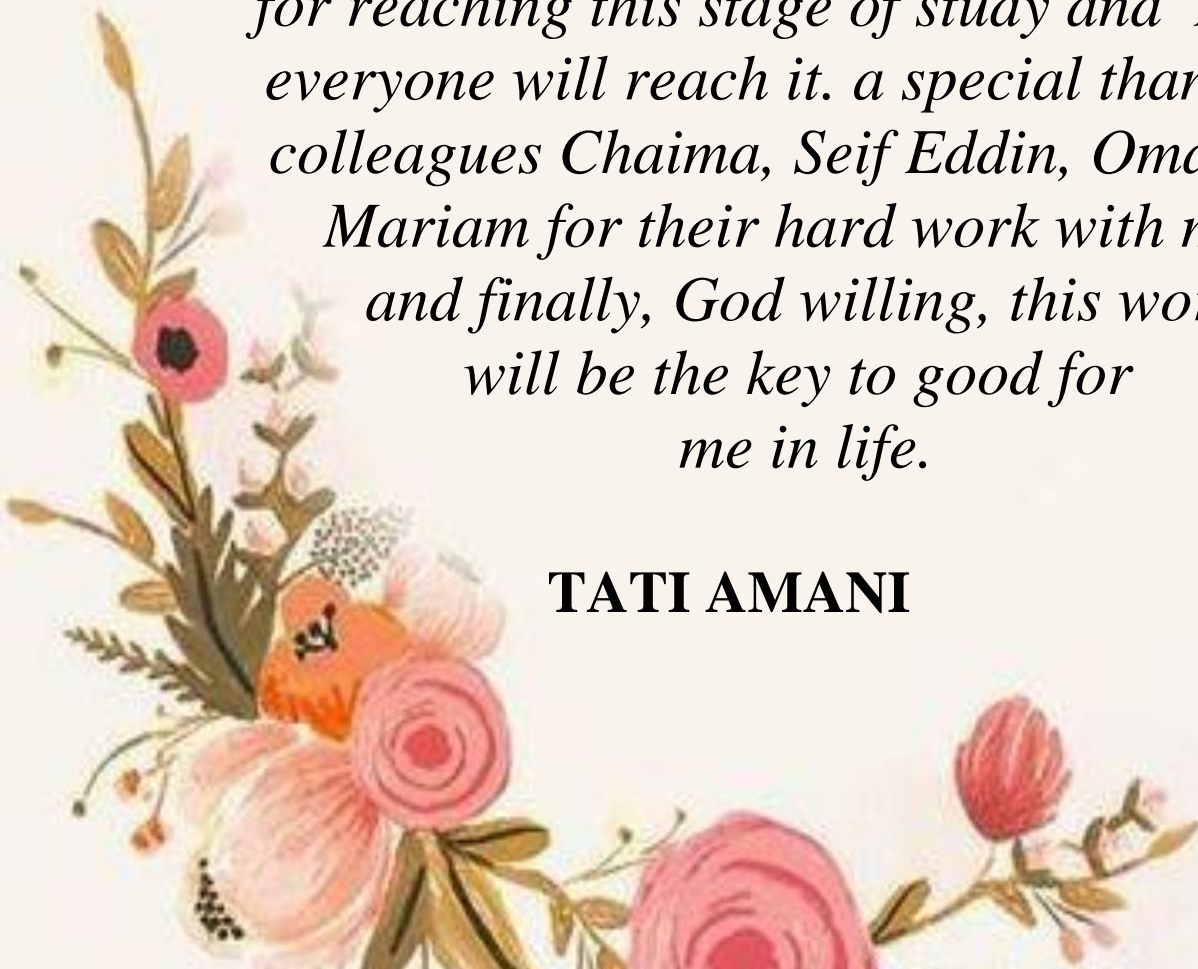




Dedication 1

I dedicate this work to my parents who worked hard with me throughout 17 years of jihad and struggle, and regardless of the amount of thanks and perseverance, I will not reach even an iota of their sacrifices for me, I ask God to grant them health and wellness Thank God for reaching this stage of study and I hope everyone will reach it. a special thank my colleagues Chaima, Seif Eddin, Omaima, Mariam for their hard work with me, and finally, God willing, this work will be the key to good for me in life.

TATI AMANI





Dedication 2

Thank God. Thank God for everything after the study process brought with it many difficulties, hardship and fatigue. I dedicate my graduation to hope for life and the secret of my success, to my dear parents, God lasts and prolongs their lives. who taught me how to hold the pen? and how to write words.

To all our relatives, loved ones and those shared by the sweet and bitter life to all my colleagues I love

BEN SACI CHAIMA

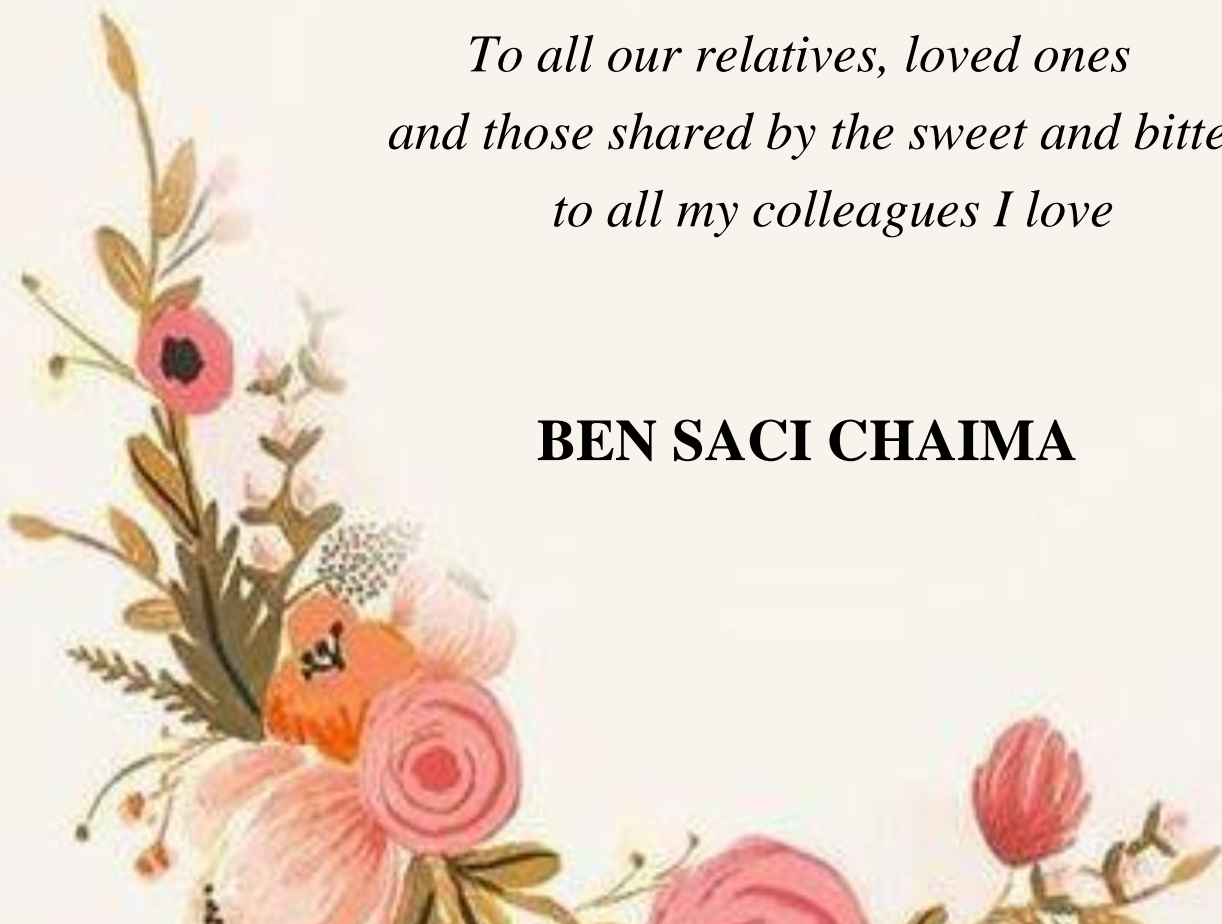


Table of content

Thanks.....	2
Dedications	3
List of figures.....	7
List of tables.....	8
List abrivation.....	10
Abstract.....	11
Itroduction general.....	14

Chapre 1 : Internet Of Things

I.HISTORY.....	17
II.Introduction.....	17
III.Definition of IOT.....	17
IV.Architectures of IOT.....	18
1)Three-layer architecture.....	19
2)Five-layer architecture.....	19
V.Domains of IoT.....	19
1)Logistics and transportation domain.....	19
2)Healthcare domain.....	19
3) Agriculture domain.....	20
4)Futuristic applications domain.....	20
5)Domain of smart environments.....	20
6)Smart energy-domain.....	20
7)Smart industry-domain.....	20
VI.Protocols of IOT	21
□Protocols of application layer.....	21
1)Constrained Application Protocol (COAP)	21
2)Message Queue Telemetry Transport (MQTT)	22
3)Extensible Message and Presence Protocol (XMPP)	22
4)Advanced Message Queuing Protocol (AMQP)	23
5)Representational State Transfer (REST)	23
VII.Characteristics of IOT.....	24
1.Interoperability.....	24
2.Scalability.....	24
3.Network environment.....	25
4.Performance.....	25
5.Reliability.....	25
6.Security.....	25
VIII.Conclusion.....	26

Chapre 2 : CoAP and HTTP

I.INTRODUCTION.....	28
II.CoAP protocol(Constrained Application Protocol)	28
1)Definition of CoAP	28
2)Features of COAP.....	28
3)COAP architecture.....	29
I.Layer Message.....	29
II.Request/Response Layer model.....	30
III.COAP Message Format.....	33
4).Example of communication COAP.....	35
5).Protocol characteristics.....	36
6)Evaluation theorique CoAP.....	38
7)Applications of CoAP.....	40
III.HTTP protocol.....	43
1)Definition HTTP.....	43
2)Features of HTTP.....	43
3)Architecture of HTTP.....	44
4)Methods of HTTP (operations)	45
5)Message exchange patterns.....	45
IV.Theoretical comparison study (HTTP and CoAP)	47
V.CONCLUSION.....	49

Chapre 3 : Simuation of CoAP protocol

I.Introduction.....	51
II.Evaluation methods.....	51
III.Methodology.....	52
IV.Tools of simulation.....	52
V.Simulation Environment.....	55
VII.CoAP simulation using COOJA.....	56
VIII.Performances evaluation experiments.....	61
IX.Conclusion.....	65

Chapre 4 : Performance and evaluation

I.Introduction.....	67
II.Standards of Evaluation.....	67
III.Performance evaluation method.....	68
IV.Evaluation results.....	73
V.Conclusion.....	76
References.....	77

List of figures

❖ Image 1: Internet of Things.....	19
❖ Image 2: IOT Architectures.....	19
❖ Image 3: COAP Protocol.....	23
❖ Image 4 : MQTT Protocol.....	23
❖ Image 5: XMPP Protocol.....	24
❖ Image 6: AMQP Protocol.....	24
❖ Image 7 Abstract Layer of CoAP	30
❖ Image 8 : Reliable Message Transport (ACK)	31
❖ Image 9 : Un reliable Message Transport (RST)	31
❖ Image 10: The successful and failure response results of GET method.....	33
❖ Image 11: A Get request with a separate response.....	33
❖ Image 12: Non confirmable request and response.....	34
❖ Image 13: CoAP message format	34
❖ Image 14 : Exemple de communication client / serveur COAP.....	37
❖ Image 15: Caching in CoAP.....	38
❖ Image 16: HTTP architecture.	45
❖ Image 17: Request message of HTTP.....	47
❖ Image 18: Reponses message of HTTP.....	47
❖ Image 19: Request-Reply model, for example: COAP and http.....	49
❖ Image20: COOJA simulator Interface.....	54
❖ Image21: Smart City components.....	57
❖ Image 22: VMware windo.....	58
❖ Image 23: Contiki OS window.....	58
❖ Image 24: Cooja simulator.....	59
❖ Image 25: COOJA window.....	59
❖ Image 26: Border Router Mote.....	60
❖ Image 27: CoAP server motes.....	60
❖ Image 28: CoAP client mote.....	61
❖ Image 29: Border router and CoAP servers.....	61
❖ Image 30: PyCharm interface.....	62
❖ Image 31: Simulation of 2 Servers in Random Topology.....	63
❖ Image 32: Simulation of 15 Servers in Random Topology.....	63
❖ Image 33: Simulation of 35 Servers in Random Topology.....	66
❖ Image 34:Simulation of 2 Servers in Linear Topology.....	64
❖ Image 35: Simulation of 15 Servers in Linear Topology.....	65
❖ Image 36: Simulation of 35 Servers in Linear Topology.....	65
❖ Image 37: Curves of energy consumption (j)against the number of CoAP server in 1 h	70

- ❖ **Image 38: Curves of energy consumption (j) against the number of CoAP server in 15 m**
.....70
- ❖ **Image 39: Curves of packet loss against the number of CoAP server in 1 h**
.....71
- ❖ **Image 40: Curves of packet loss against the number of CoAP server in 15 m**
.....71
- ❖ **Image 41: Curves of energy consumption (j) against the number of CoAP server in 1 h**
.....72
- ❖ **Image 42: Curves of energy consumption (j) against the number of CoAP server in 15 m**
.....72
- ❖ **Image 43: Curves of packet loss against the number of CoAP server in 1 h.....73**
- ❖ **Image 44: Curves of packet loss against the number of CoAP server in 15 m73**

List of tables

- Table 1: Protocols of IOT layers.....22**
- Table2: comparison between different IOT application layer protocols25**
- Table 3: CoAP methods.....32**
- Table 4: CoAP response codes.....32**
- Table 5: CoAP defines several options used on both request and responses.....36**
- Table6: Practical Applications of CoAP Protocol.....44**
- Table7: HTTP methods.....46**
- Table8: Comparison between HTTP and CoAP.....50**
- Table 9: Simulation environment.....56**

List of abbreviations

- **IEEE** Institute of Electrical and Electronics Engineers
- **IETF** International Engineering Task Force
- **CoAP** Constrained Application Protocol
- **HTTP** Hypertext Transfer Protocol
- **IoT** Internet Of Thing
- **M2M** Machine to Machine
- **MQTT** MQ Telemetry Transport
- **NFC** Near Fields Communication
- **RFID** Radio Frequency Identification IP Internet Protocol
- **IPV6** Internet Protocol version 6
- **IPV4**Internet Protocol version 4
- **WSN** Wireless Sensor Networks
- **AMQP** Advanced Message Queuing Protocol
- **MQTT** Message queue telemetry transport
- **XMPP** Extensible messaging and presence protocol
- **JMC** Java message service
- **TCP/IP** Transmission Control Protocol / Internet Protocol
- **UDP** User Datagram Protocol
- **Wi-Fi** wireless fidelity
- **RPL** routing protocol for low power and loss network
- **ETSI** européen télécommunication standards institue
- **DDS** Data Distribution service
- **ApiRest** representational state transfer
- **6Lowpan** over low power wireless personal area network
- **LPWAN** low power wireless link
- **BLE** Bluetooth a base consummation
- **LTE** Long term evolution
- **Rest** representational state transfer
- **Soap** simple object access protocol
- **XML** Extensible markup language
- **OMG** Objectives management group
- **DCPS** district of Columbia schools
- **Uri** Uniform resource. Identifier
- **LAN** Local Area Network

Abstract:

The Internet of Things (IOT) is a type of network that connects an infinite number of smart devices over the Internet using different protocols, through sensors to perform data exchange and connections for tracking, monitoring, management and other things.

In the coming years, the Internet of Things is expected to become one of the most prevalent technologies and requires major changes in Internet protocols to suit its requirements.

Because smart devices are small and constrained, they do not provide effective connectivity. Since the application layer controls the quality of communication, its protocols ensure the reliability of work and communication. Among these protocols, Internet Things devices are often used by COAP because it is the most convenient and solves the most problems.

In this project, we provide a glimpse into the performance of the most commonly used application layer protocols: COAP, by evaluating power consumption, lost packets by simulating the protocol using Cooja emulator in Contiki OS, studying the results obtained from the simulation and then analyzing them to find out Where is the strength of this protocol.

Keywords: IoT, Application protocols, CoAP, Performances evaluation, Simulation, Cooja Simulator.

الملخص

إنترنت الأشياء (IOT) هو نوع من الشبكات التي تربط عددًا لا حصر له من الأجهزة الذكية عبر الإنترنت باستخدام بروتوكولات مختلفة، من خلال أجهزة الاستشعار لأداء تبادل البيانات والاتصالات للتعقب والمراقبة والإدارة وأشياء أخرى.

من المتوقع أن تصبح إنترنت الأشياء في السنوات القادمة واحدة من أكثر التقنيات انتشارًا وتتطلب تغييرات كبيرة في بروتوكولات الإنترنت لتلائم متطلباتها.

نظرًا لأن الأجهزة الذكية صغيرة ومقيدة، فإنها لا توفر اتصالاً فعالاً. نظرًا لأن طبقة التطبيق تتحكم في جودة الاتصال، فإن بروتوكولاتها تضمن موثوقية العمل والتواصل. من بين هذه البروتوكولات، غالبًا ما تستخدم أجهزة COAP أجهزة Internet Things لأنها الأكثر ملاءمة وتحل معظم المشكلات.

في هذا المشروع، نقدم لمحة عن أداء أكثر بروتوكولات طبقة التطبيق شيوعًا: COAP، من خلال تقييم: استهلاك الطاقة، الحزم المفقودة عن طريق محاكاة البروتوكول باستخدام محاكي Cooja في Contiki OS، ودراسة النتائج التي تم الحصول عليها من المحاكاة ثم تحليلها لمعرفة أين تكمن قوة هذا البروتوكول.

الكلمات المفتاحية: إنترنت الأشياء، بروتوكولات التطبيق، CoAP، تقييم الأداء، المحاكاة، محاكي Cooja.

Resumé

L'Internet des objets (IOT) est un type de réseau qui connecte un nombre infini d'appareils intelligents sur Internet en utilisant différents protocoles, via des capteurs pour effectuer l'échange de données et des connexions pour le suivi, la surveillance, la gestion et d'autres choses.

Dans les années à venir, l'Internet des objets devrait devenir l'une des technologies les plus répandues et nécessitera des changements majeurs dans les protocoles Internet pour répondre à ses besoins.

Parce que les appareils intelligents sont petits et limités, ils ne fournissent pas une connectivité efficace. Puisque la couche application contrôle la qualité de la communication, ses protocoles assurent la fiabilité du travail et de la communication. Parmi ces protocoles, les appareils Internet Things sont souvent utilisés par COAP car ils sont les plus pratiques et résolvent le plus de problèmes.

Dans ce projet, nous donnons un aperçu des performances des protocoles de couche application les plus couramment utilisés : COAP, en évaluant : la consommation d'énergie, les paquets perdus en simulant le protocole à l'aide de l'émulateur Cooja dans Contiki OS, en étudiant les résultats obtenus à partir de la simulation, puis les analyser pour savoir où est la force de ce protocole.

Mots clés : IoT, Protocoles applicatifs, CoAP, Evaluation des performances, Simulation, Cooja Simulator.

General Introduction

Anything connected to the Internet is a smart thing, from here came the definition of smart devices, which are electrical devices connected to the Internet through sensors to expand their capabilities without any human intervention, that is, linking physical elements of data, and this is called the Internet of Things (IOT), which allows the creation of Wide range of applications in several fields such as transportation, healthcare, smart environments (home, work, factory), agriculture, etc.

Because smart devices are small and constrained, they do not provide effective communication. The protocol is one of the most important communication factor in the Internet world, and since the application layer controls the quality of communication, its protocols ensure the reliability of work and communication. Among these protocols, we find CoAP, MQTT, HTTP, XMPP..... etc. Internet devices often uses Constrained Applications Protocol (COAP) because it is one of the most suitable and respects many criteria of IoT.

In order to use this technology, IoT applications must consider all its characteristics including reliability, interoperability, scalability, and security. However, these characteristics are difficult to guarantee due to the nature of the connected devices and the design of complex networks in the Internet of Things.

There are three methods for assessing CoAP: analysis method, real experiment, simulation, among which we chose the simulation. In this work, we decided to study the CoAP protocol by simulation with the Cooja simulator, in the Contiki Os software. This allows us to run simulations under a network with low bandwidth, high latency and packet loss, in order to verify that the system is running efficiently and respecting the required characteristics. In this study we conducted two experiments: the first is the number of servers in a random topology and the second is the number of servers in a linear topology with increasing number of servers each time. In these experiments, we consider two parameters: power consumption and packet loss.

Through the measurements obtained from the simulation, we can gain insight into the effectiveness of the protocol within a restricted wireless network, through the results of the experiments.

With the measurements obtained from the simulations which are energy consumption and packet loss, we can analyse the effectiveness of the protocol within a restricted wireless network (WSN), through the results of the experiments.

This work was arranged as follows:

In the first chapter we published the concept of IoT, some of its applications journals, architectures, properties, and some protocols where we focus specifically on application layer protocols, and finally we mentioned some advantages and disadvantages of IoT.

In the second chapter, we theoretically studied the CoAP and http protocols, and explained some of their characteristics and advantages, the architectural structure, and ways of working, and a general theoretical comparison between them.

In Chapter Three, we discussed how to simulate CoAP in a COOJA simulator and study its effectiveness through some experiments based on the smart home scenario, according to different criteria.

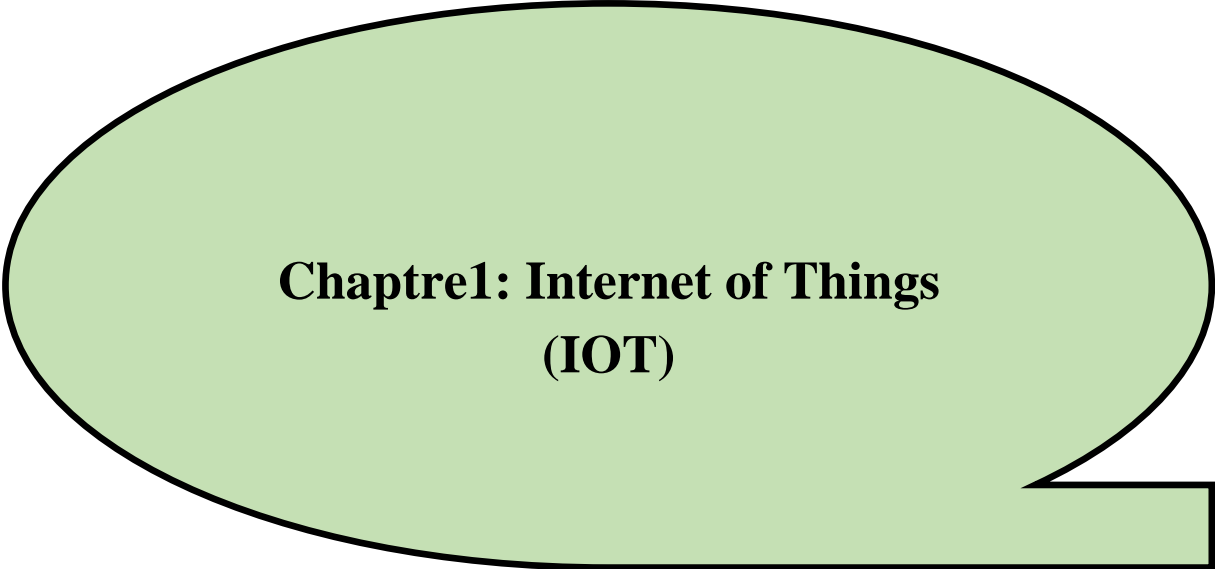
In the fourth chapter, we will evaluate the performance of CoAP with aspects of IoT through the results obtained from simulations

RELATED WORK:

Due to its efficacy and degree of use in the last years, many works have studied and compared the performance of application CoAP protocol in the IoT environment. We now review some relevant of these works:

- In This paper [35], analyzes the performance offered by one of the most popular application layer protocols in the Internet of Things: Constrained Application Protocol (CoAP). This analysis aims to examine the features and capabilities of this protocol and determine its feasibility for operating under restricted devices using security support.
- In this paper [36], he compared protocols such as CoAP, 6LoWPAN, and RPL using the Contiki OS COOJA simulator. This work aims to analyze these protocols based on some criteria like power consumption, radio duty cycle, average time between beams, etc. It was analyzed and concluded that each protocol should be preferably based on its application path. However, depending on the power consumption or the average time between packs, CoAP produces a slightly better result.

- In this paper [39], this paper summarizes some wireless protocols. Then it introduces CoAP and corresponding security protocol DTLS.
- In this paper [40], trade-offs between performance, power consumption, and level of security are explored for the most recent version of the widely accepted Contiki OS (version 3.x) when IETF-supported DTLS security is enabled for the constrained Application Protocol (CoAP). More specifically, the DTLS framework is integrated into the Contiki 3 CoAP stack for two different cipher sets, and performance is evaluated against insecure CoAP implementations through simulation, in terms of speed, overall memory, and power consumption for different WSN server network environment.
- In this paper [49], reviewed XMPP, AMQP, CoAP, MQTT, DDS, and MQTT-SN protocols that are available in the application layer of IoT and afterward they compared every protocol with knowing their execution. To assess their performance, they had picked different measurements, for example, packet transmission ratio, throughput, power consumption, and bandwidth. It is audited that the MQTT, XMPP, AMQP, and MQTT-SN protocols that keep running on TCP produces higher PDR while contrasted with CoAP and DDS protocols that keep running on UDP, which does not back retransmission of packets. Also, it is watched that CoAP has higher throughput, consistent ideal bandwidth utilization, and low power consumption differentiated with other data protocol that is an appropriate real-time environment. After that, they watched, how the gadget gets managed remotely utilizing Contiki OS with COOJA simulator.



**Chaptre1: Internet of Things
(IOT)**

I.HISTORY:

Kevin Ashton first described the Internet of Things as an emerging global information service architecture based on the Internet in a 1999 presentation. It refers to connected objects that can be identified by unique addressing and are able to interact with one another and collaborate with their neighbors to achieve common goals. [1][2]

In the context of object identification and tracking, members of the same MIT group defined the Internet of Things in 2001 as "an intelligent infrastructure that connects objects, information, and people through a computer network."

Defining the Internet of Things in terms of worldwide RFID technology and standardized coordination and support: a global network architecture that connects physical and virtual things through data capture and communication capabilities. [3]

II.Introduction:

Internet of Things technologies, which represent the next evolution of the Internet and are among the most pervasive topics in every field due to the massive change they have brought about for the entire society, including we, have been enabled by the rapid development of mobile Internet, micro computing, and machine-to-machine (M2M) communication. It is a network infrastructure that connects physical objects to the Internet and allows them to be remotely sensed and controlled; it also allows them to act intelligently to communicate with the real world; and it also allows smart environments to identify objects and retrieve information to achieve common goals.[4] [3]

III.Definition of IOT:

The Internet of Things (IoT) is a global open and dynamic architecture that consists of a network of "smart" objects that connect and communicate over the Internet and gather and share data via embedded software, cameras, and sensors that detect light, sound, and motion; where smart devices operate automatically, or are controlled and monitored remotely. [6].

In the Internet of Things, an object could be a person with a pacemaker, a farm animal wearing a chip (transmitter), a car with sensors to alert the driver when the tire pressure is too low, or any natural or man-made object that can be set IP address and the ability to transmit data over the network.

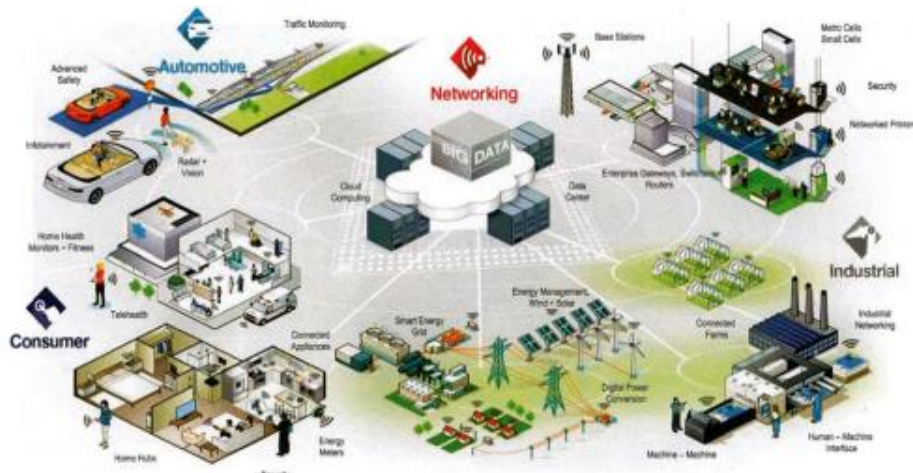


Image 1: Internet of Things

IV. Architectures of IOT:

An architecture is a framework that defines the physical elements, technical layout, network configuration, operating procedures, and data formats used. The structure of the Internet of Things varies depending on the operating time. [8]

For future development, IoT systems need robust design and standards. Two types of structures have been proposed for IOT, one three-layer and the other five-layered; The European Telecommunications Standards Institute (ETSI) has developed a three-layer architecture for the Internet of Things: the perception layer, the network layer, and the application layer. [9] The five-layer structure consists of a three-layer structure in addition to two layers of processing and works. As shown in the following figure:

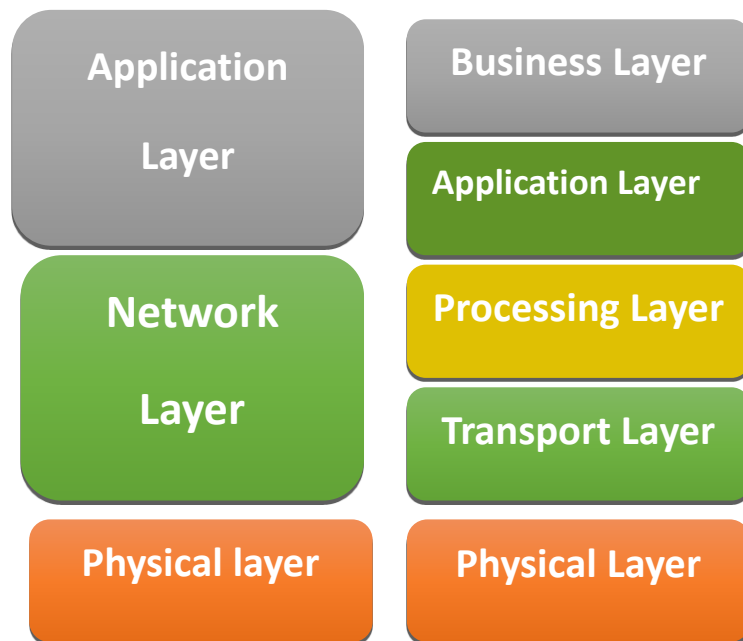


Image 2: IOT Architectures

1) **Three-layer architecture:** [8]

It refers to three levels: perception, network, and application.

1. Layer of Perception: A physical layer with sensors for sensing and obtaining information about the environment; it senses certain physical parameters or recognizes other intelligent objects in the surrounding environment.

2. Network Layer: Responsible for connecting smart devices, network devices and servers. Its capabilities are also used to transmit and process sensor data.

3. Application layer: Responsible for providing the user with services related to the applications. It defines many applications of the Internet of Things, such as smart homes, smart cities, smart health, and others.

2) **Five-layer architecture:** [8]

It includes the three layers (perception, network, and application) in addition to the processing and business layers.

1. **The transport layer:** Transfers sensor data from the perception layer to the processing layer via Wi-Fi, 3G, LAN, Bluetooth, RFID and NFC...

2. **The processing layer (middleware):** The data from the transport layer is stored, analyzed and processed by the processing layer; using many technologies such as databases, cloud computing and data processing units.

3. **The business layer:** Responsible for overseeing the entire IoT ecosystem, including applications, business models, profits, and user privacy.

V.Domains of IoT:

The potential applications of the Internet of Things are many and varied and permeate all areas of the daily lives of individuals, organizations and society as a whole. [7]; these can be grouped into the following domains:

1) **Logistics and transportation domain:**

Sensors and processing power are increasingly being integrated into advanced cars, trains, buses and bicycles, as well as roads and/or rails. Roads and goods transported are also equipped with sensors to improve traffic routing, assist in driving, manage warehouses and monitor goods.

2) **Healthcare domain:**

By automatically collecting and detecting data, IoT technologies in healthcare ensure clinical signs of patients to facilitate remote monitoring; Monitor medical freezers containing

vaccines and medicines, as well as dental equipment such as toothbrushes. And monitor physical activity. [10] [7].

3) **Agriculture domain:**

The Internet of Things (IoT) will be the primary tool for crop environment monitoring and pollution control, with the most powerful sensors currently connected to the cloud via cellular/satellite networks [], allowing us to see data in real time.

It has facilitated agriculture in many services, including greenhouses, animal husbandry and tracking, aeration study on farms, and management of agricultural fields, including control of fertilization, electricity and irrigation... [7].

4) **Futuristic applications domain:**

Many of the applications presented have been developed or can be realized in the future because the underlying technologies are currently available, and thus we may envisage many applications that have not yet emerged or whose implementation is still very complex [12].

5) **Domain of smart environments:**

A smart environment, whether it is an office, home, city, industrial facility or leisure place, makes “work” simple and fun thanks to the intelligence of things.

Among the activities of the Internet of Things in the environment we find control of water, electricity and gas; smart lighting; remote controls; weather and flood monitoring; and life protection. One of the most important advantages is that these systems will aid the elderly and disabled in the community. [7]

6) **Smart energy-domain:**

Customers use smart meters to analyze energy from various sources such as wind turbines, and smart energy is represented in the monitoring and management of energy consumption as well as its distribution. It also provides photovoltaic energy services, such as monitoring and optimizing the performance of a solar power plant. [7]

7) **Smart industry-domain:**

Deployment of IoT in industry will be of great help to the economy and service sector, as it will allow tracking of the entire product and distribution chain through supply chain monitoring, as well as detection of gas levels and leaks in industrial environments, and maintenance and repair services. Product traceability will also aid in the fight against counterfeiting, fraud and other economic crimes.

VI.Protocols of IOT :

According to the Internet of Things' three-layer architecture, the physical layer transmits packets between network segments, the transport layer establishes communication channels for data transmission and use by the application layer, and the application layer is the most important layer, which includes the protocols used by users to provide services or exchange data, among which the protocol considered in this note is included (CoAP Protocol).

Application	Transport	Network	Data binding	Physical
AMQP	TCP	IP(ipv6,ipv4,...)	IEEE 802.15.4	BLE
CoAP	UDP	6LoWPAN	LPWAN	Ethernet
DDS		Icmp		LTE
MQTT		Rpl		NFC
XMPP				RFID
API REST				Wi-Fi/802.11
WebSocket				ZigBee
				Z-Wave

Table 1: Protocols of IOT layers

❖ Protocols of application layer:

In an IoT environment, the application layer provides data distribution and communication between devices. Application layer protocols are the messaging protocols that devices use to transfer data over the Internet.

The current IOT application layer protocols are CoAP, XMPP, MQTT, DDS, AMQP, REST, WebSocket, and JMS.

1) Constrained Application Protocol (COAP):

It is an application layer protocol for IoT applications; [13][14] It is based on UDP and includes a lightweight reliability mechanism.

REST-based CoAP was designed by the IETF Constrained RESTful Environment Working Group to provide a lightweight RESTful (HTTP) interface. [15][16] Allows clients and servers to expose and consume web services such as Simple Object Access Protocol (SOAP) [13] [14]. CoAP aims to enable low-power sensors to use REST services while respecting their power limitations. [15][16].

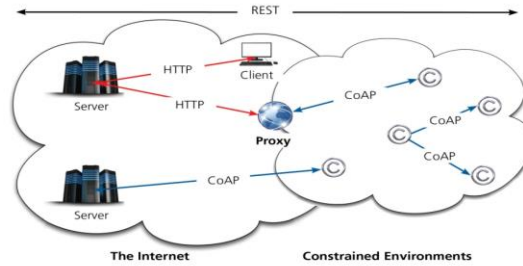


Image 3: COAP Protocol

2) Message Queue Telemetry Transport (MQTT):

The MQTT protocol was introduced by IBM and standardized by OASIS [18, 15]. It is an open, lightweight and easy-to-implement protocol designed to provide embedded communication between applications and middleware on the one hand, and networks and communications on the other. It follows a publishing/subscription structure, as it consists of three main components: publishers, subscribers, and a medium.

The MQTT is used in a variety of areas, including healthcare, monitoring, and energy measurement. As a result, small devices with limited power and memory can be routed in vulnerable locations and networks with poor bandwidth [13].

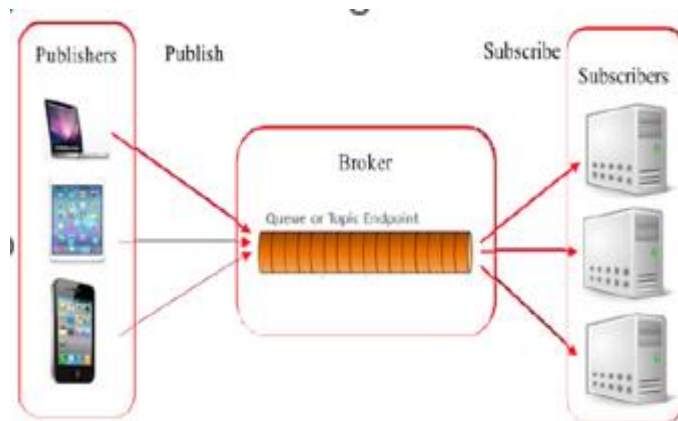


Image 4 : MQTT Protocol

3) Extensible Message and Presence Protocol (XMPP):

It is a family of real-time messaging technologies that includes its XML Stream technology [19]; it is designed for chat and messaging applications, such as multi-party chats and audio and video calls. Standardized by the IETF, it supports both a publish/subscribe architecture and a request/response architecture. It allows adding new applications but does not guarantee QoS, which makes it unsuitable for M2M communications. Although XMPP is not widely used in IoT, it has received a lot of attention to improve its architecture and support [19, 15].

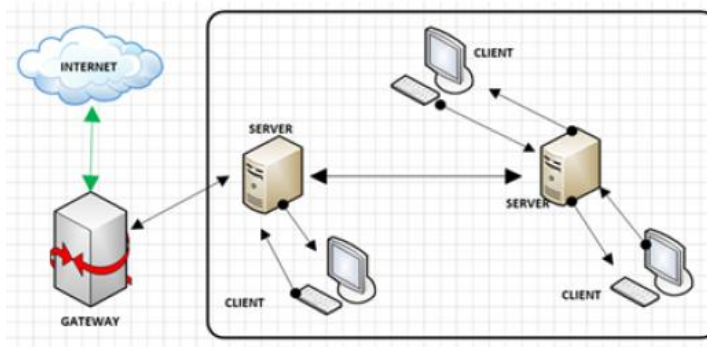


Image 5: XMPP Protocol

4) Advanced Message Queuing Protocol (AMQP):

It is an open standard protocol that focuses on message-based environments. It allows reliable communication and works over TCP, with a post/subscription architecture like MQTT. Communication is handled by two main components: message exchange and queues, i.e., receiving and delivering messages from publishers to queues. Queues indicate topics that subscribers are interested in and who will receive sensory data as soon as it becomes available [20].

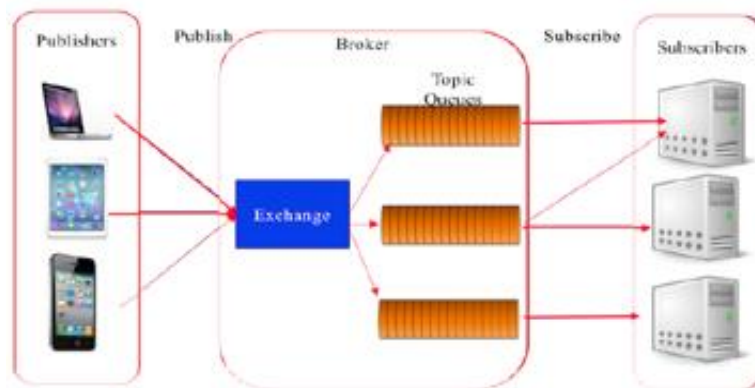


Image 6: AMQP Protocol

5) Representational State Transfer (REST):

Roy Fielding is a REST presenter. Developed by W3C Technical Engineering Group. REST aims to improve component scalability and autonomy while reducing network connections and latency.

A RESTful service is a REST-based service. It is mostly used to create web services that are easy to maintain, lightweight and scalable. REST uses the available HTTP methods. RESTful services should have the following features: representations, URIs, unified interface, statelessness, resource bindings, and caching, but they come at a high cost and power in lightweight IoT applications, and there are no QoS options. [22]

Table2: comparison between different IOT application layer protocols [22]

S. N O	Protocol	Transport	Architecture	Security	Qos	Mode of message exchange	Responsiveness
1	CoAP	UDP	Request/Response	DTLS	YES	Synchronous/ Asynchronous	Real/Time
2	MQTT	TCP	Publish/Subscribe	TLS/SSL	YES	Asynchronous	Real/Time
3	AMQP	TCP	Publish/Subscribe	TLS/SSL	YES	Asynchronous	Non-Real/Time
4	XMPP	TCP	Request/Response Publish/Subscribe	TLS/SSL	NO	Synchronous/ Asynchronous	Real/Time
6	REST	HTTP	Request/Response	HTTPS TLS/SSL	NO	Synchronous	Non-Real/Time

VII.Characteristics of IOT:

Internet of things consists of several aspects, including [41,42,43,44,45,46]:

1. Interoperability:

Since networks are made up of many heterogeneous devices and standards, the ability to communicate proportionally to their differences is essential for the Internet of Things. You can see interoperability problems from different points of view due to heterogeneity, one of which is the interoperability of a device with sufficient computing resources and capabilities like Raspberry Pi and smartphones. Network interoperability that deals with mechanisms to enable the seamless exchange of messages between systems through different networks must therefore address issues of addressing, routing, resource optimization, security, quality of service, and mobility support. The level of syntactic interoperability is important to enable migration, seamless messaging between different Internet of Things (IoT) systems has been suggested to provide greater interoperability. Semantic interoperability allows communication parties to share information and interoperability with the platform. It is therefore essential that the devices and protocols be able to accurately interpret the exchange of messages. Lack of system interoperability can increase complexity and cost.

2. Scalability:

The number of gadgets has exceeded and continues to outpace the world population because of the expansion in global population. As a result, successful routing systems for Wireless Sensor Networks (WSNs) participating on the Internet of Things must be scalable and adaptable to changes in network architecture. Hence, the scalable protocol should perform well as the network grows or the workload increases. The scalability of sensor networks also

supports the expansion of the network to more nodes that might not be expected during the initial network design phase.

3. Network environment:

The type of network and the type of nodes that make up the network can have a big impact on whether an IoT protocol succeeds or fails. When building up a network, selecting the appropriate protocol to meet the needs of the devices makes a significant impact in the environment.

4. Performance:

The speed and response time capability of IoT protocols are determined by the effectiveness of the protocols and the methods used to handle problems on the network and devices.

5. Reliability:

The primary objective of collecting big data for IoT systems is to provide real-time situational awareness with reliable data. As a result, if users want to improve the use of the application, the exchange of data between sensors must be reliable. A trusted protocol, for example, in computer networks, is a communication protocol that tells the sender whether the delivery of data to the intended recipients has been successful. RDP (Reliable Data Protocol), for example, uses an efficient and trustworthy data transmission service to download and fix problems. The primary purpose of RDP is to remain effective in contexts where message segments are sent in non-sequential order, or where there are long delays and transmission failures.

One of its drawbacks of IOT is the average life expectancy of the resource, it is important that IoT protocols do not deplete battery life too quickly.

6. Security

Some of the data exchanged by IoT devices may contain sensitive information that should not be widely accessed, such as webcams that allow a homeowner to remotely monitor their possessions from their smartphone or a sensor that acts as a key to unlocking your home or car. The owner is the only one who has access. Data sharing is a characteristic of the Internet of Things application layer. Data access and authentication, phishing attacks, malware activity, layer security requirements, and malware attacks are among the most common application category security challenges.

VIII.Conclusion:

For the purpose of introducing and explaining the Internet of Things, in this chapter we presented the definition of the Internet of Things, and some domains that use its applications, tools and architecture and some of the most famous protocols where we focus on application protocols, and then we passed to present their characteristics, advantages and disadvantages.

In the next chapter, we will present a theoretical study of CoAP and HTTP and compare them.



Chapter 2: CoAP and HTTP

I. INTRODUCTION:

IoT protocols are divided into several parts of the application layer protocols, transport, network and infrastructure, in the application layer there are several protocols such as CoAP, MQTT, AMQP, AXMP, XMPP, HTTP Rest ... etc.

The most widely used protocol is CoAP (Constrained Application Protocol), which was designed by the IETF for use in M2M IoT applications and is suitable for constrained devices with limited resources. CoAP is based on HTTP Rest (Hypertext Transfer Protocol) which is a client-server communication protocol developed for the web that can run on any trusted connection.

In this chapter, we will present CoAP and HTTP protocols, we will give their characteristics, and how they work ... finally, we will present the main differences between the two protocols.

II. CoAP protocol(Constrained Application Protocol):

1) Definition of CoAP :

Constrained Application Protocol (CoAP), a simple and inexpensive protocol based on asynchronous exchange of messages based UDP, defined by the IETF's CoRE Working Group in RFC 7252, has a lightweight mechanism to provide reliability.

Designed specifically to manage simple resources and devices on a network of constrained nodes and for applications of M2M machines such as smart energy and building automation. It is very similar to HTTP and is intended to work on equipment with very limited resources.

2) Features of COAP:

The main features of CoAP: [3]

- ❖ Constrained web protocol fulfilling M2M requirements.
- ❖ Asynchronous message exchanges.
- ❖ UDP binding with optional reliability supporting unicast and multicast requests.
- ❖ Low header overhead and parsing complexity.
- ❖ Simple proxy and caching capabilities.
- ❖ URI and Content-type support.
- ❖ Very low and simple cost to analyze.
- ❖ A stateless HTTP mapping, allowing proxies to be built providing access to CoAP resources.
- ❖ Security binding to Datagram

Transport Layer Security (DTLS)

3) COAP architecture:

CoAP interactive model is like HTTP's client/server model. Fig 13 shows that CoAP employs a two layers structure. The bottom layer is Message layer that has been designed to deal with UDP and asynchronous switching. The request/response layer concerns communication method and deal with request/response message [34].

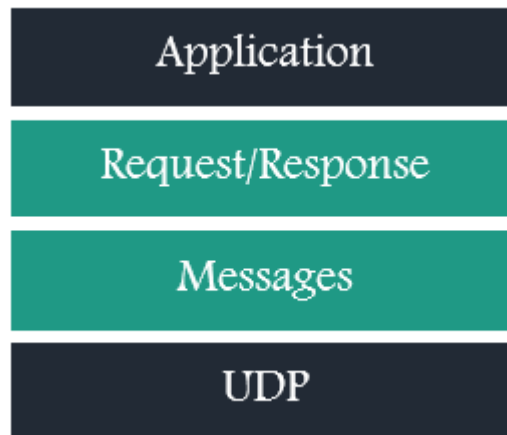


Image 7 Abstract Layer of CoAP

I.Layer Message:

The messaging model of CoAP is based on the exchange of compact messages over User Datagram Protocol (UDP) between endpoints. Messages are transported, by default, over UDP. However, CoAP can also be used over DTLS and other transports such as TCP, SMS or SCTP. Messages are shared by requests and responses. Each message has a message ID used to detect duplicated messages and for optional reliability.

CoAP defines four types of messages:

1. **Confirmable (CON):** the confirmable message requires an acknowledgement response from the receiver to provide a reliability functionality.
2. **Non-Confirmable (NON):** the non-confirmable message does not require an acknowledgement from the receiver.
3. **Acknowledgement (ACK):** acknowledges the confirmable message.
4. **Reset (RST):** the reset is used instead of ACK in case that CON or NON cannot be processed.

A) Reliable message transport (confirmable message) :

Keep retransmission until get ACK with the same message ID (like 0x8c56 in fig. 14). Using default time out and decreasing counting time exponentially when transmitting CON. If recipient fail to process message, its responses by replacing ACK with RST. Fig. 14 shows a reliable message transport.

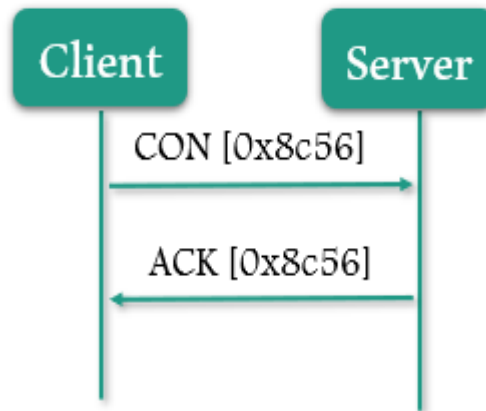


Image 8 : Reliable Message Transport (ACK)

B) Unreliable message transport (non-confirmable message) :

Transporting with NON type message. It does not need to be asked but must contain message ID for supervising in case of retransmission. If recipient fail to process message, server replies to RST. Fig. 15 shows unreliable message transport.

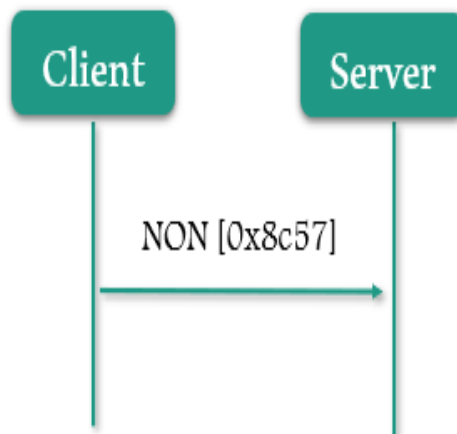


Image 9 : Un reliable Message Transport (RST)

II. Request/Response Layer model:

1. Request methods:

The client requests an action using a Method code on a resource which identified by Uniform Resource Identifier (URI) on a server. The CoAP defines four different methods :

Method	Description
GET	Retrieves information from a specified resource identified by the requested URI.
POST	Submits information to be processed to a specified resource. The output result depends on the target resource, usually, results in the target resource being created or updated
PUT	Requests that the resource identified by the URI be created or updated with the carried information representation.
DELET	Requests that the identified resource be deleted.

Table 3: CoAP methods

2. Response Codes:

The server sends back to the client a response code indicates the outcome of the request process.

There are three classes of Response code:

- **Success 2.xx:** This class indicates that the request has been successfully received and processed. Here are some examples of response codes under this class: 2.01 Created, 2.02 Deleted.
- **Client Error 4.xx:** This class indicates that the request from the client was not valid or has an error. For example, 4.00 for a Bad request or 4.04 Not found this response code is like the common HTTP 404 which indicate that the request is correct, but the server could not find the resource identified by the URI.
- **Server Error 5.xx:** This class indicates that the server has an error or incapable of processing the request. For example, 5.03 Service unavailable.

Code	Beschreibung	Code	Beschreibung
2.01	Created	4.05	Method Not Allowed
2.02	Deleted	4.06	Not Acceptable
2.03	Valid	4.12	Precondition Failed
2.04	Changed	4.13	Request Entity Too Large
2.05	Content	4.15	Unsupported Content-Format
4.00	Bad Request	5.00	Internal Server Error
4.01	Unauthorized	5.01	Not Implemented
4.02	Bad Option	5.02	Bad Gateway
4.03	Forbidden	5.03	Service Unavailable
4.04	Not Found	5.04	Gateway Timeout
		5.05	Proxying Not Supported

Table 4: CoAP response codes

➤ A request is carried in a Confirmable or Non-confirmable message, the response to the request has different scenarios:

1. Piggy-backed response: Client sends request using CON type or NON type message and receives response ACK with confirmable message immediately. In fig10, for successful response, ACK contain response message (identify by using token), for failure response, ACK contain failure response code.

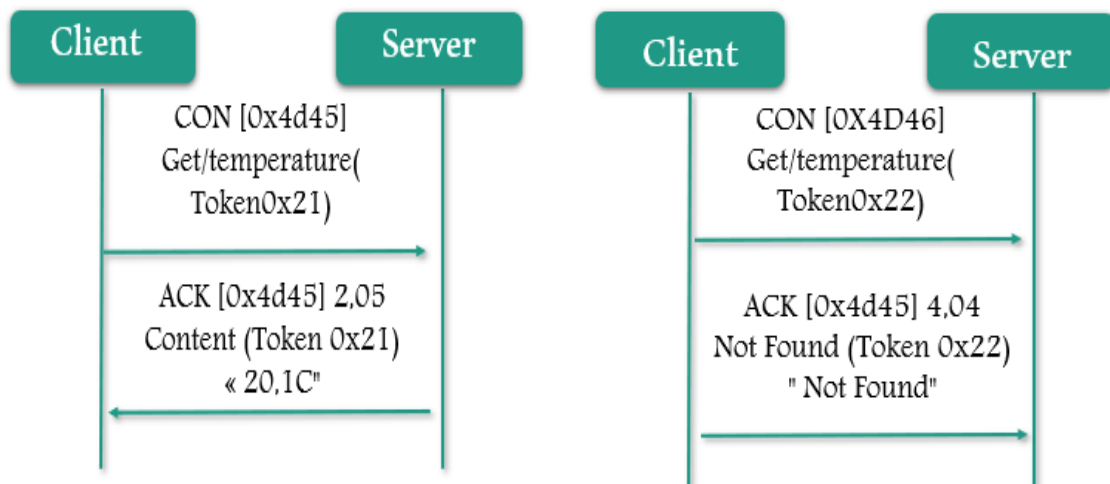


Image 10: The successful and failure response results of GET method

2. Separate response: If server receive a CON type of message but not able to response this request immediately, it will send an empty ACK in case of client resend this message. When server ready to response this request, it will send a new CON to client and client reply to a confirmable message with acknowledgment. ACK is just to confirm CON message, no matter CON message carry request or response (fig. 11).

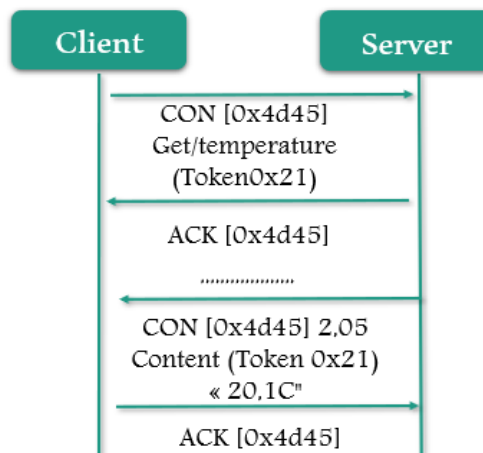


Image 11: A Get request with a separate response

3. **Non confirmable request and response:** unlike Piggy-backed response carry confirmable message, in Non confirmable request client send NON type message indicate that Server do not need to confirm. Server will resend a NON type of message with response (fig. 12).

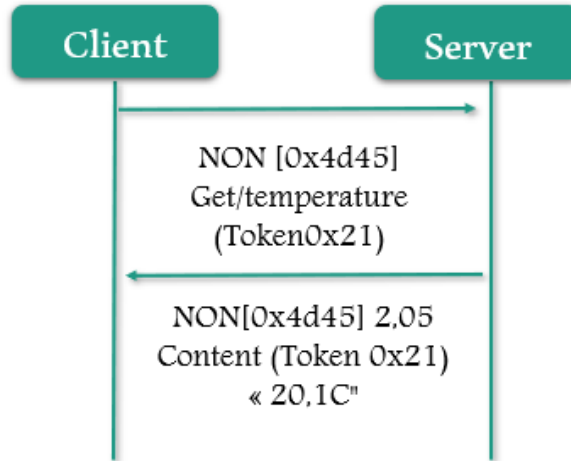


Image 12: Non confirmable request and response

III.COAP Message Format

The format of the messages of the protocol has been designed to be simple and light to reduce the typical overhead caused by the headers of the protocols. All the messages start with a fixed-size 4-byte header, which is mandatory. Then, it is followed by a variable-length Token value, a sequence of zero or more CoAP Options in Type-Length-Value (TLV) format and an optional payload [35].

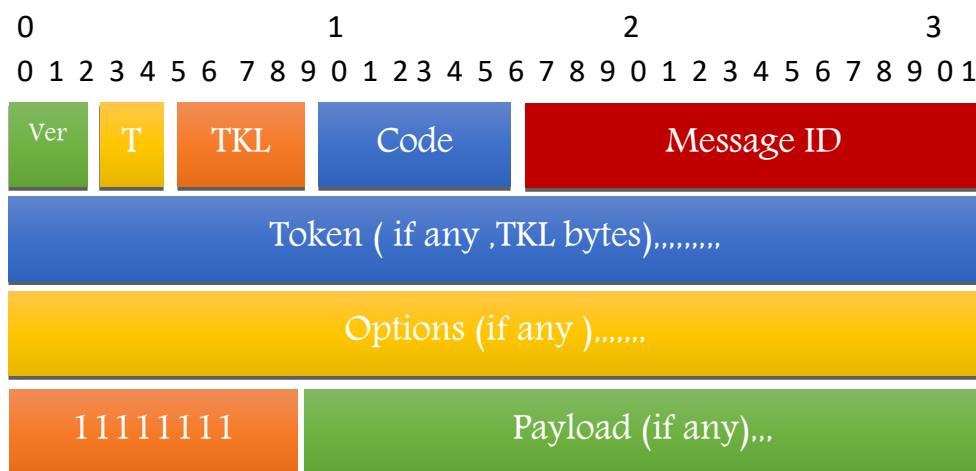


Image 13: CoAP message format

The message header bits are defined as follows:

➤ **Version (Ver):** a 2-bit integer that shows the CoAP version used.

➤ **Type (T):** 2-bit unsigned integer.

Indicates if this message is of type Confirmable (0), non-confirmable (1), Acknowledgement (2), or Reset (3).

➤ **Token Length (TKL)** determines the length of the variable-length token field.

➤ **Code :** is an 8-bit unsigned integer, the Request message (1-10) or Response message (40-255).

➤ **Message ID :** a 16-bit integer used for reliable transmission, duplicate detection and to match ACK/RST to corresponding CON/NON messages.

➤ **Payload:** The payload has a set of one byte that is called the Payload Maker, which marks the start of the payload data. If the Payload Maker has a value of all ones (0xFF16), there is data present, otherwise, the payload is empty.

➤ **Options:** In the options field, if any, it must contain the option number, option value length, and the value itself. The option number is not declared specifically, rather is calculated by the equation: option number = option delta + previous option number.

-Option delta is used to establish the difference between current option number from its previous one.

-Option length simply indicate the size of the option value.

-Option value is a sequence of 'L' length, define by the length field and can contain the following formats: empty (zero), opaque, unit (option length) or string(UTF-8).

The options field has two different classes that define the way unrecognized options are to be handled by endpoints. Those classes are either critical or elective.

Option	Rôle
Content Format	determines the representation format of the message payload.
ETag	or entity-tag defines a resource-local identifier that can distinguish between similar representation of the same resource that vary over time.
Location-Path and Location-Query:	defines an absolute path, a query string or both that specifies the new location and/or query argument of a resource created with a POST request.
Max-Age	indicates a maximum time in which the response cached can be considered fresh.
Proxy-Uri	it refers a request to a forward-proxy. The forward-proxy is requested to forward the request or service it from a valid cache and return the response
Proxy-Scheme	it is used to assemble an absolute-URI in a proxy request. The absolute-URI is constructed from Uri-Host, Uri-Port, Uri-Path, Uri-Query options
Uri-Host	determines the Internet host of the resource being requested.
Uri-Path	determines one segment of the absolute path to the resource
Uri-Port	defines transport-layer port number of the resource.
Uri-Query	specifies one argument to describe the resource.
Accepte	clients can use this option to indicate the acceptable content format to get from a response.
If-Match	it may be used to make a request conditional on the current existence or value of an ETag for one or more representations of the target resource.
If-None-Match	opposite to If-Match, If-None-Match may be used to make a request conditional on the non-existing target resource. Auxiliary to PUT request to avoid any accidental resource overwrite, especially if the same resource is being used by multiple clients.
Size1	mainly used in block-wise transfer, it determines the resource representation size in a request.

Table 5: CoAP defines several options used on both request and responses

4. Example of communication COAP:

To better understand how COAP works, we will look at an example of client-server communication. Figure 20 illustrates this: [14]

- **Step 1:** The client sends a GET request with an id = 123 of the confirmable CON type and with a uri-quer y = /light.

The Client then awaits the destination server's acknowledgement response. If the client does not receive a response by the end of time T, it assumes the packet has been lost and sends the message at a random time interval to avoid network congestion. At the end of MAX –Retransmitted, the server will be considered inaccessible.

- **Step 2:** The server sends a 2.00 OK with the same id as the client, of type ACK acknowledgement, and the response to the request as payload (not visible here).

- **Step 3:** The client sends a GET request with an id = 124 of confirmable type CON and with a uri-query = /humidity.
- **Step 4:** Because the server did not react in a timely manner, the client retransmits the request. There was a break in the action.
- **Step 5:** The server responds with a 2.00 OK with the same id as the client, of type ACK acknowledgement and the response to the request as payload (not visible here).

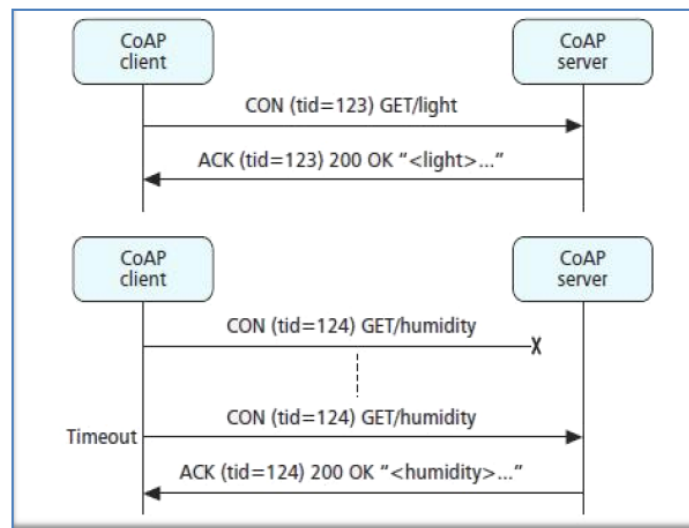


Image 14 : Exemple de communication client / serveur COAP

5. Protocol characteristics:

a) Proxying:

The COAP proxy is intended to support applications that must communicate with WSN nodes, such as smart city development.

A Constrained Application Protocol (CoAP) proxy allows the adaptation of web application protocol packets and CoAP devices. The proxy acts as an intermediary, relaying and/or forwarding data between endpoints. A proxy improves network performance, grants access to dormant devices, and reduces power consumption, bandwidth, and network traffic. [48]

There are two types of proxies in the CoRE architecture: a routing proxy selected by the client, and a reverse proxy selected by the origin server. As a result, two devices that implement both protocols can be easily connected.

- ✓ **Proxying CoAP-HTTP:** Since CoAP methods are equivalent to HTTP methods, it allows access to HTTP server resources for CoAP clients, HTTP, TCP and optionally TLS can be created easily.[49]
- ✓ **HTTP-CoAP Proxying:** allows access to the resources of a CoAP server for HTTP clients.

the client must specify the absolute path to the resource including the schema (CoAP / CoAPs) in the method invocation for to send an HTTP request to the proxy. Once the proxy has delivered the message, it will request the specified CoAP resource[49].

b) Caching:

In CoAP, the purpose of caching is to reuse a previous response message to satisfy the current request.

When the response is "recent", it can be used to complete subsequent requests without connecting to the original server, improving efficiency and reducing latency and network trips.

When the endpoint contains one or more stored responses to a GET request but cannot be used, the ETag option can be used on the GET request to give the original server an opportunity to select a stored response to use and update, even if it is a new request; The "verification" mechanism is used for this. [49]

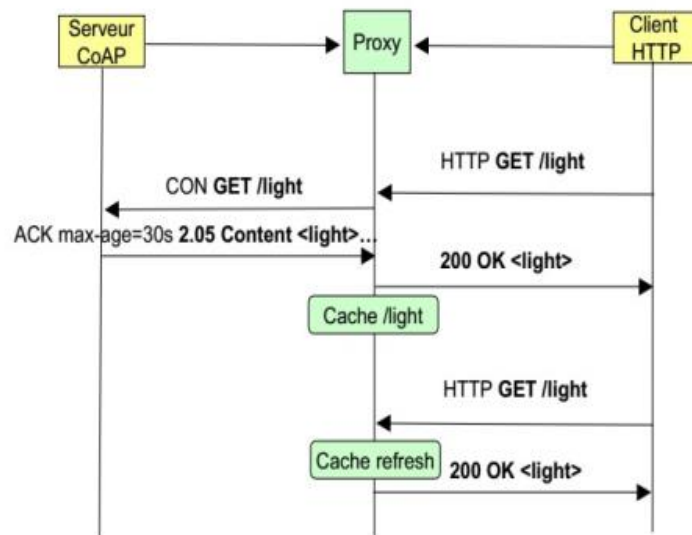


Image 15: Caching in CoAP

c) Resource Discovery:

In machine-to-machine (M2M) applications it is important to find a way to discover restricted server resources.

The resource guide contains descriptions of server resources that enable customers to find all the resources they need in a single application. The device must know how to access the DR to use it for recording or research.

On the other hand, direct resource discovery is difficult in many M2M environments because nodes may sleep for long periods of time. To solve this problem, CoRE resource directories (RD) can be used, which host resource descriptions on other servers. The CoAP server can then register

its resources with one or more RDs in this way that clients can find these resources by using the RD to perform searches. [51][14]

d) Securing CoAP:

Datagram TLS (DTLS) is used over UDP to secure CoAP. In addition, DTLS avoids the cryptography overhead issues that plague lower-layer security protocols. DTLS solves two problems reordering and packet loss. [49].

It's also crucial to remember that when utilizing DTLS for client-server communication, the client request must include a matching session, timestamp, and message ID, and the same goes for the answer.

6) Evaluation theorique CoAP:

When weighing the benefits and drawbacks of a protocol, there are numerous aspects to consider. The following are a few of them that will be explored here: [48]

1. Network Environment:

The suitability of a protocol is often determined by the environment and the nodes in the network. The type of network and its requirements may have an impact on the overall performance of the protocol, especially if all the requirements are met. CoAP is a low-power network standard that is compatible with both 6LoWPAN and LLN and seems ideal for use on short-range networks.

Although CoAP enhances the RESTful architecture and enables some HTTP functionality for restricted devices, it is still a new protocol, so other protocols may take precedence. On the other hand, CoAP has a bright future as a major protocol in the development of the Internet of Things

2. Performance:

CoAP and HTTP use a typical request/response mechanism, clients request resources, and the server responds to those requests. On the other hand, polling may be harmful to a restricted network if the client requires an existing representation of the resources. Recursive polling is used in HTTP to solve this problem; Clients send request messages to the server, and the server answers only when the updated version is available. For long-lived devices, this approach reduces latency and improves processing and network resources. However, this is not the best option for limited hardware. In general, adding complexity and costs may slightly reduce performance.

To ensure network stability and limit the number of messages flowing and nodes, CoAP uses a congestion control mechanism, and the transmission of unverified messages can bypass the network.

3. Energy Consumption:

Since many IOT devices lack reliable power sources and depend on batteries, one of the problems they face is developing a system that consumes less power while still maintaining performance and capabilities. Power availability varies in a network of constrained nodes, and they will have one of the following types of power constraints:

- **Limited Energy Event:** Specifies how much energy is used for a single event, such as pressing a button.
- **Limited energy period:** They have access to maintenance, such as replacing the battery, or they have a way to harvest energy.
- **Lifetime Power Limited:** Provides a limited amount of power over its life cycle because there is no mechanism to recharge or replace the main battery.

Power management is an issue in the physical layers and the MAC layer. The emergence of smart objects has necessitated modifications to reduce energy use, such as CoAP, CoAP provides a number of applications that help reduce energy consumption, and proxies and caching can be used to reduce response time and energy consumption.

4. Cost Efficiency:

The main benefits of reducing network costs are the deployment of standard protocol solutions, such as CoAP, where a non-standard solution breaks the principle of the Internet. Abolishing the end-to-end principle will lead to the need to translate from the standard IP protocol to the proprietary protocol in meters.

As a result, application portals have been introduced which not only add to the complexity of the network but also take time and money to manage, install and run.

CoAP uses the CoRE/Resource Discovery link service to facilitate the low-cost addition, replacement and scaling of new devices as devices can discover resources and services to connect themselves to the network.

5. Interoperability:

The IETF CoRE Working Group has made significant efforts to standardize CoAP standards. Interoperability is made possible by the standardization of the Internet Protocol in IoT ecosystems.

The main advantage of making CoAP open source is that it helps fix the protocol's maturity problem by allowing IoT developers and industry to experiment with the protocol. However, the drawback is that alternative implementations of the protocol are accessible, which makes interoperability impossible to guarantee. Likewise, the open-source protocol provides a

powerful tool for code transparency and supports changes that must be made in the code to support interoperability.

6. **Scalability:**

CoAP has various scalability characteristics that distinguish it from other protocols and make it a better fit for the Internet of Things. CoAP makes it easy to find devices, services, and resources. The CoRE link format discovery method allows for a flexible approach that allows device mobility and scalability. It is easy to replace or add nodes to the network as nodes can perform an automated process that allows them to find devices and other resources.

7. **Reliability:**

The trusted protocol provides notification of the data delivery status, and with UDP adopted as the primary transport protocol, CoAP reliability is provided through verifiable messages that require acknowledgment in return. However, while this approach ensures that the message reaches the intended recipient, it does not provide any indication that the message was delivered successfully and without errors. Therefore, in CoAP, reliability is low and optional; Some connections may be flagged as untrustworthy and need not be acknowledged, but CoAP uses the message identifier in both types of messages to avoid duplicates.

8. **Security:**

CoAP is vulnerable to Internet attacks such as DoS, eavesdropping and phishing. However, CoAP offers several security modes through its own DTLS binding. One of the main challenges with constrained devices is that security may not be a high priority, and therefore may not be implemented in all cases because message slowness takes priority.

DTLS provides reasonable transport layer security, but it also has some drawbacks: First, some DTLS cipher suites feature initial handshake operations that add complexity and burden to restricted devices. Again, there is no clear definition of DTLS for multicast communications. There are ideas aimed at improving the implementation of CoAP security, some refer to security in multicast connections, others point to ways to reduce the DTLS overhead, and still others look at other security protocols such as IPsec.

7) Applications of CoAP:

CoAP has been used in many areas to provide various services such as interoperability, integration, authentication, authorization, streaming, etc. Using CoAP has helped us establish a secure and efficient connection to the device. This section provides an overview of some CoAP applications and services. [55]

✓ **Interoperability and Integration:** CoAP is used for interoperability with other application layer protocols like HTTP and MQTT, as well as the integration of different healthcare standards. [55]

a) **Interoperability with HTTP:** HTTP Interoperability: It focuses on making mobile healthcare platforms compatible with the Internet. Creates a prototype of a CoAP-based smartphone agent that enables interoperability of medical sensors as well as RESTful interaction with other nodes.[59]

b) **CoAP Interoperability with MQTT for Healthcare:** Although the use of MQTT in IoT devices has the benefit of facilitating interpersonal communication, the restricted rest time is a negative. As a result, CoAP is used to create a messaging system based on the MQTT architecture.

[59]

c) **Interoperability with Healthcare Platform:** CoAP is used to build and implement a global healthcare platform with respect to IEEE 11073 PHD, a healthcare standard for medical devices and system interoperability. The IOT device includes a CoAP server, a healthcare data collector, and a state manager. The proposed approach enables IoT environments to use the ISO/IEEE 11073 DIM/Service Model in conjunction with CoAP. [60].

d) **Integration of Healthcare Standards:** CoAP is used to explain the design and implementation of a communications system. To communicate between medical IoT devices such as medical sensors attached to patients and hospitals, CoAP is used to integrate two healthcare standards: ISO/IEEE 11073 and IHE PCD-01. The authors evaluated the performance of CoAP in this application to that of MQTT and HTTP. The results show that CoAP allows for faster transactions while using fewer internet resources. [61]

e) **Integration with OSGP:** Open Smart Network Protocol (OSGP) is a communication protocol for exchanging data between Smart Network (SG) devices and does not support integration with limited CoAP IoT devices. It offers a solution called CoAP and OSGP Integration to map data packets between CoAP and OSGP (COIIoT). [62].

✓ **Security :** In any application, security is a critical issue. Security services such as authentication, authorization and access control are implemented using CoAP. [55]

a) **Authentication and Authorization:** CoAP has been used to combine authentication, authorization, and accounting infrastructure with Extensible Authentication Protocol (EAP), to overcome the limitations of Low Power Wide Area Networks (LP-WAN). Low-cost CoAP-EAP (LO-CoAP-EAP) is a proposed method for solving network access authentication for limited devices in LP-WAN. [62]

b) Authentication and Access Control: CoAP authentication and access control framework for the Internet of Things, such as Kerberos with CoAP, have been proposed, and improved ECDSA is released for cryptography and privacy. A ticketing-based solution is provided for authentication and access to services. [63]

✓ **Streaming Services:** Due to CoAP's mass transfer capability, it can be used in streaming services such as media or video streaming. [55]

a) Media Streaming services: Dynamic Broadcasting via CoAP (DASCo) is proposed. The algorithm uses Dynamic Stream over HTTP (DASH) formats and uses CoAP to deliver media clips to consumers. CoAP's Pause and Wait transfer can help download media clips. Moving a block is useful for determining the progress of a download and knowing when it stopped. [64]

b) Video Streaming Services: The proposed technology adjusts the video quality based on the available transfer rate, download time of the clip, and the size of the next video. The CoAP client determines the video rate for each segment from the range of video rates available on the CoAP server in order to achieve high-quality streaming. [65]

c) Cloud Computing Services: Cloud integration with IoT extends its reach, as CoAP implements data transfer between IoT sensor nodes and the cloud. Humidity and temperature sensors are used in the system and their data is sent in JavaScript Object Notation (JSON) to the Things Board cloud endpoint using CoAP POST messages. [66]

✓ **Resource Observation and Discovery** mentions the role of CoAP in implementing resource monitoring and discovery processes in an IoT environment and WSNs. [55]

a) Resource Observation: In IoT and WSN networks, CoAP has been used to monitor resources. The temperature is monitored, for example, by a temperature sensor in a simple application that acts as a CoAP server, allowing any CoAP client to obtain resource information. [67]

b) Resource Discovery: A mixed strategy is used to find resources, which is based on the proactive resource directory discovery (RD) mechanism. The presence of an RD in the network is announced in a PDR using CoAP POST messages, which nodes can store. [68,69,70]

✓ **Real-Time Remote Monitoring:** CoAP was used to implement a real-time telehealth monitoring system. [55]

The patient's vitals can be monitored and displayed in real time using a web browser that acts as a CoAP client and sensors connected to the patient's body act as a CoAP server.[71]

✓ CoAP has also been successfully implemented in other areas, including:

Domain industrial	logistics and product life management	Purchases
		Fastpayment
		Identify the equipment
Domain health and well-being	medical health care	Altamatmedical
		Monitor medicalequipment
		Smart hospital services
domain smart city	public safety and environmental monitoring	Video surveillance
		emergency plan
		Monitor employees
	smart homes and buildings	Lighting
		Energy management
		child protection

Table6: Practical Applications of CoAP Protocol[58]

III.III.HTTP protocol:

1) Definition HTTP:

Hypertext Transfer Protocol is a client server communication protocol developed for the web. It is an application that can run on any trusted connection, HTTP uses TCP as the default transport protocol and TLS/SSL for security. The HTTP server defaults to port 80. Typical HTTP clients are web browsers that allow the user to directly access server data one by one via a URI, a text-based protocol that does not specify header size and message payloads, but rather depends on the web server or programming technology. [57] HTTP is a globally accepted web messaging standard that provides many features such as persistent connections, suckers, and segmented transport encryption. [26]

2) Features of HTTP: [27]

There are three primary features that make HTTP simple and powerful:

- **HTTP is media independent:** It specifies that any type of media content can be sent by HTTP if both the server and the client can handle the data content.
- **HTTP is connectionless:** It is a connectionless approach where the HTTP client i.e. the browser starts the HTTP request and after the request is sent, the client disconnects the connection from the server and waits for the response.

- **HTTP is stateless:** The client and server are only aware of each other during the current request. After that, they both forget each other. Neither the client nor the server can keep information about the various requests across web pages.

3) **Architecture of HTTP:** [28]

The HTTP protocol is a request/response protocol based on the client/server-based architecture where web browsers, robots and search engines, etc. act like HTTP clients, and the Web server acts as a server.

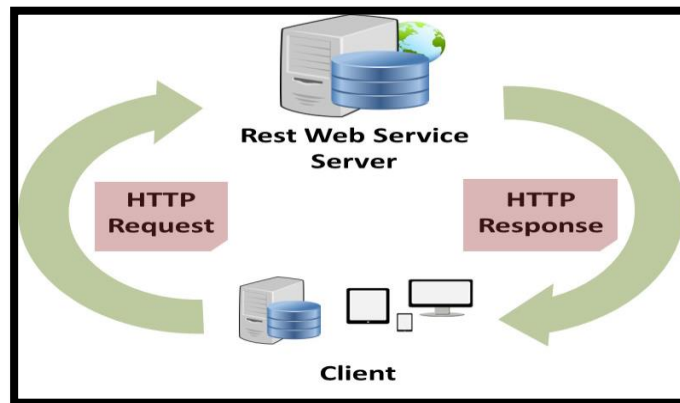


Image 16: HTTP architecture.

❖ **Client:**

The HTTP client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and body content over a TCP/IP connection.

❖ **Server:**

The HTTP server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible entity-body content.

4) Methods of HTTP (operations):[27]

HTTP Request	Description
GET	Asks to get the resource at the requested URL
POST	Asks the server to accept the body info attached. It is like GET request with extra info sent with the request.
HEAD	Asks for only the header part of whatever a GET would return. Just like GET but with no body.
TRACE	Asks for the loopback of the request message, for testing or troubleshooting.
PUT	Says to put the enclosed info (the body) at the requested URL.
DELETE	Says to delete the resource at the requested URL.
OPTIONS	Asks for a list of the HTTP methods to which the thing at the request URL can respond

Table7: HTTP methods**5) Message exchange patterns:**

There are two types of HTTP messages, requests and responses, each with its own format.

1. An exchange begins when a client sends a request message
2. When the request message is received, the server responds with a response message.
3. The exchange ends when the customer has received the response, or the connection is interrupted.

I. Requests Messages:

An example HTTP request:

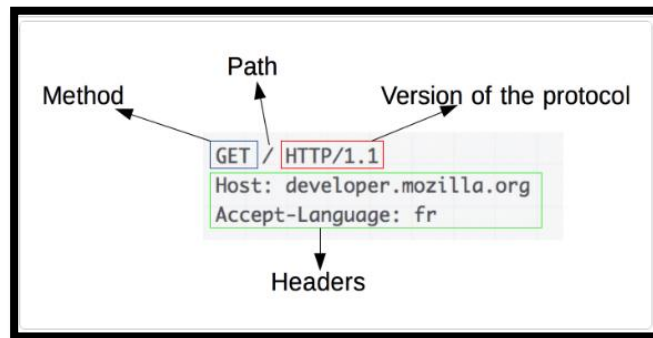


Image 17: Request message of HTTP

Requests consists of the following elements:

I. HTTP method, usually a verb like **GET** or **POST** or a name like **OPTIONS** or **HEAD** that specifies the operation the client wants to perform.

- The path of the resource to fetch; the URL of the resource stripped from elements that are obvious from the context, for example without the [protocol](#) (`http://`), the [domain](#) (here, `developer.mozilla.org`), or the TCP [port](#) (here, `80`).
- The version of the HTTP protocol.
- Optional [headers](#) that convey additional information for the servers.
- A body, for some methods like **POST**, like those in responses, which contain the resource sent.

II. Responses Messages:

An example response:

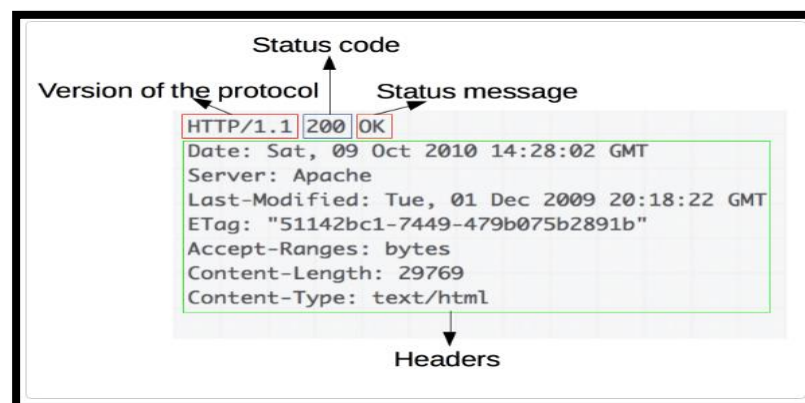


Image 18: Reponses message of HTTP

Responses consist of the following elements:

- ❖ The version of the HTTP protocol they follow.
- ❖ A [status code](#), indicating if the request was successful or not, and why.
- ❖ A status message, a non-authoritative short description of the status code.
- ❖ HTTP [headers](#), like those for requests.
- ❖ Optionally, a body containing the fetched resource [30].

IV. Theoretical comparison study (HTTP and CoAP) :

From the protocol architectures discussed in this chapter above, CoAP is based on UDP (there is a new protocol, CoAP TCP, but is still being studied), and HTTP is based on TCP. CoAP is designed for use on devices with limited (constrained) resources, while HTTP is used for unlimited devices.

CoAP and HTTP are based on the REST model which provides a request/response-based interaction architecture. However, machine-to-machine interactions usually result in CoAP implementation in both client and server roles. A CoAP request is similar to an HTTP request in that it is made by the client to request an action on the server resource, and the server responds with a response that may include a representation of the resource. With proxy components, CoAP can easily interact with HTTP, and HTTP clients can talk to CoAP servers and vice versa, allowing for better web integration and the ability to meet IoT needs.

Compared with CoAP, HTTP has more computational complexity and lower data rate, but CoAP has shorter latency and lower power consumption. Unlike HTTP, CoAP uses datagram directed transmission to handle asynchronous interactions.

If CoAP is used on a network with a firewall, UDP is not recommended because firewalls will block UDP messages because they cannot distinguish between legitimate UDP messages and spam messages. In this case, Transmission Control Protocol (TCP) can be used as an alternative transport protocol. [48]

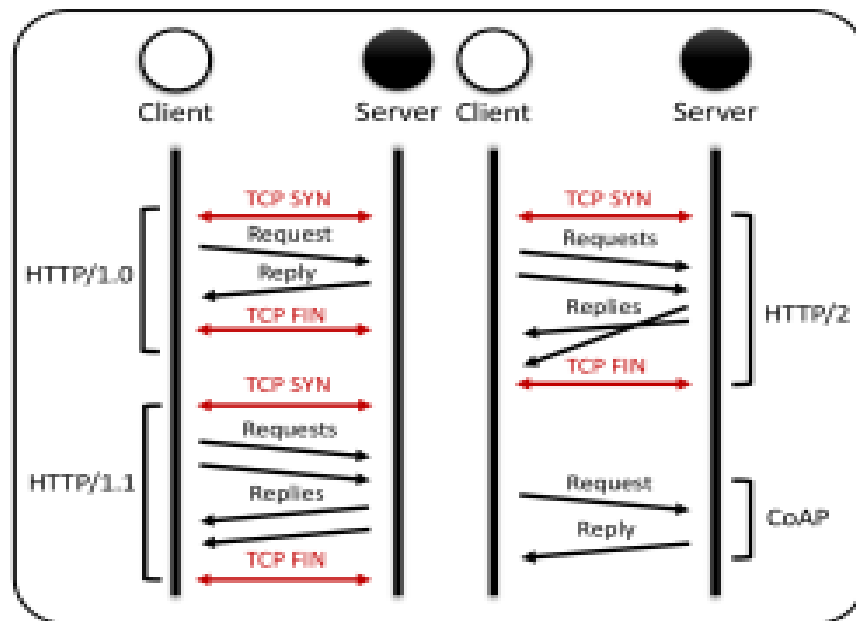


Image 19: Request-Reply model, for example: COAP and http

For the three versions of HTTP plus CoAP, image 32 provides examples of distinct client/server interactions.

1. In HTTP 1.0 a TCP connection is closed after one HTTP request/reply pair.
2. In HTTP 1.1 a survival mechanism was introduced, whereby a TCP connection can be reused to send multiple requests to the server without waiting for a response.
3. HTTP 2.0 provides a multiplexing method and responses can be received asynchronously over a single TCP connection.
4. The fourth interaction is specific to CoAP, which does not rely on a TCP connection to exchange request/response messages between the client and server. On the other hand, the publishing and subscription model arose from the need to provide destinations for distributed and asynchronous data communications. Today the solution appears in the form of several Post and Subscribe (MoM) middleware.

✓ This illustrated table represents a comparison between HTTP and CoAP:

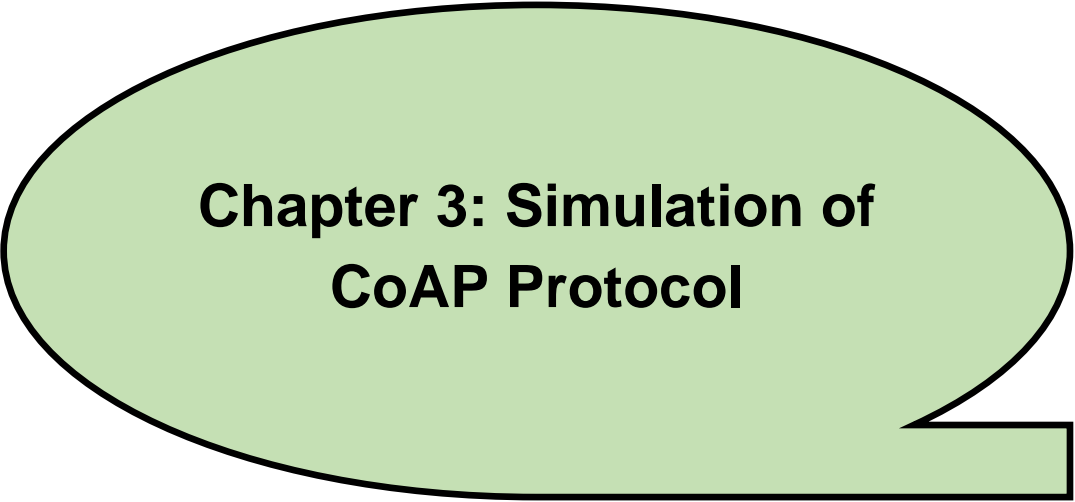
	CoAP	HTTP
Protocol	UDP	TCP
Used for	Devices limited resources (Constrained)	Devices unlimited resources(unrestricted)
Based on	REST model	REST model
Computational complexity	Low	High
Data rate	High	Low
Energy consumption	Low	High
Latency	Low	High
Exchanges msg	Asynchronously	Synchronously
Weight	Light	Heavy

Table8: Comparison between HTTP and CoAP

V.CONCLUSION:

The purpose of this chapter is to present a theoretical study of CoAP and HTTP protocol, where we presented the most important elements, including their features, structures and characteristics, and we give a theoretical comparison of the two protocols .

In the next chapter, we will present **CoAP simulation** in COOJA simulator.



**Chapter 3: Simulation of
CoAP Protocol**

I. Introduction:

Simulation consists of modeling the entire studied system and simulating it numerically using the around environment generated from real-world observations or probabilistic models [54]. Simulation has the advantage of allowing you to work with systems that are not available, for example, it is less expensive to perform an initial simulation of the alternatives under consideration during the design phase. Moreover, simulation is a very versatile way to investigate a topic, as it allows the program to resume with parameter modifications and track execution to be executed in a real environment without any interruption.

In this chapter, we will review several assessment methods before explaining how to simulate CoAP using the specific scenario (Smart City) and the tools that were used. We will change and increase the number of servers and packet loss to simulate different CoAP experiences.

II. Evaluation methods:

There are different techniques for evaluating system performance on WSN. Among them: analytical modeling, we can cite analytical modeling, measurements obtained from real experiments, or simulations.

I. Analytical methods:

It presents analytical methods for studying the behavior of a system by solving the mathematical equations on which its mathematical model is based. Analytical methods are useful for solving equations that take little time to calculate. Moreover, analytical methods allow a better knowledge of the workings of the system, as one is better able to examine some imbalances of the system by solving its model, and thus suggest modifications to solve them such as formal methods [53]. In the scientific literature, the analytical technique requires verification, model study, and quantitative measurements.

II. Real Experience:

Validating protocols and applications through real-world tests is a challenging task. In fact, studying the problem from real experiments is difficult for several reasons: reproducing experiments is difficult, external factors can disrupt experiments over which the experimenter has no control, and studying increases, decreases, and changes in the pattern of speed and motion is complex. [54] Real experiment approaches have disadvantages that require making restrictive assumptions about the real system in order to obtain viable models, and because our topics do not require real application because this is just performance research, this method was not chosen.

III. Simulation:

Discrete event simulation consists in reproducing the behavior of a system by studying a specific perception of its model. The benefit of simulation is that it provides a way to study any model as long as the simulation tool is adapted to the model under study. On the other hand, it has the disadvantage of requiring a lot of automated computing time [72].

There are many separate event simulators. Among them we mention the NS-2, NS-3 and OMNeT network simulators, which allow simulation of different types of networks, including queuing networks, as well as OPNET and COOJA is a tool for performance analysis.

COOJA, being the default network emulator for Contiki, was originally compiled with Contiki 3.0. COOJA provides an easy-to-use interface that allows for quick simulation and analysis setup. While our topic requires performance study, COOJA has proven to be one of the best tools for protocol simulation due to its flexibility, scalability, and rapid prototyping.

III. **Methodology:**

We will use the smart city scenario to evaluate the performance of CoAP. We will use the COOJA simulator and PyCharm to simulate various scenarios by changing the variables (number of servers with changing the topology). In order to get the metrics results as curves to facilitate evaluation and analysis, we have changed some files in the Contiki system using language C.

IV. **Tools of simulation:**

To simulate the COAP protocol, we must use several tools:

A. **Software tools:**

✓ **VMware:** To run simulations in Cooja Simulator, there are two ways to install Contiki OS: either by downloading and installing the file directly into Ubinto (LUNIX) or in Windows by installing VMware.

Due to the numerous and insecure versions of LUNIX, we worked directly for VMware which is an American computer company founded in 1998 affiliated with EMC Corporation, which offers many proprietary products related to virtualization for x86 architectures. It is also by extension the name of a group of virtualization software. [5]

✓ **Contiki OS:** Contiki is an open source, lightweight operating system for networked and sensor connected systems. It includes IP connectivity for both IPv4 and IPv6, as well as RPL, 6LoWPAN, CoAP, and other functions [47]. Contiki offers three simulation environments: Cooja, MSPsim and Netsim.

Recently developed by a Swedish research team, Contiki has been a huge success and attracts more developers. Contiki are used in many commercial and non-commercial systems, such as city sound monitoring, streetlights, etc.

It is designed for small IoT devices with limited memory, power, bandwidth, and computational capabilities. Contiki only takes a few kilobytes to implement and can fit into an entire operating system in less than 30 kilobytes. [56]

✓ **COOJA:** It is a wireless sensor network simulator based on Contiki OS. It is a versatile Java-based emulator that enables application programs to be written in C using the native Java interface. One of the useful features of COOJA Emulator is that it can copy application programs in both high-level algorithm development and low-level driver development at the same time.

Without changing the core COOJA code, application developers can change sections of the simulation environment. This means that the system can be introduced to new elements such as interfaces, plug-ins and radio media, or existing parts can be modified. We may run variance simulation with various conditions and system parameters, such as different packet generation rates, different MAC protocols, and different network architecture, using the advantages of COOJA. [52]

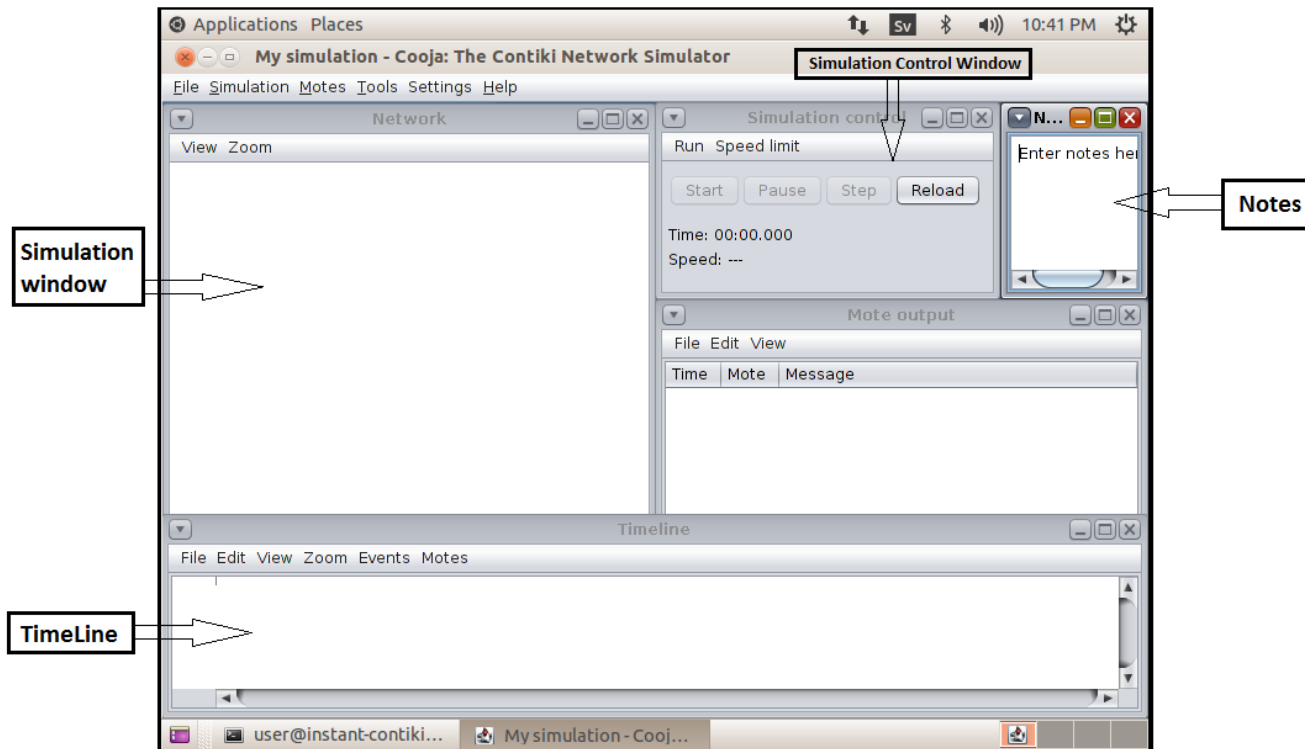


Image 20: COOJA simulator Interface

❖ **Simulation window:**

In a simulation we have several windows:

❖ The Timeline window: is located at the bottom of the screen, displays all the events of communication in the simulation over time, making it easy to comprehend what is going on in the network.

- ❖ The Network window: is located at the top left of the screen, shows us all the nodes in the simulated network.
- ❖ The Mote Output window: is located on the right side of the screen, shows us all the serial port prints from all nodes.
- ❖ The Notes window :is located at the top right is where we can put notes for our simulation.
- ❖ The Simulation control window is where we can start, pause and load our simulation.
- ✓ **PyCharm:** is an integrated development environment used for programming in the Python language.

It allows code analysis and has a graphical debugger. It also allows unit test management, version control software integration, and supports web development with Django.

Developed by the Czech company JetBrains, it is a cross-platform software that runs on Windows, Mac OS X, and GNU/Linux. It is available in a professional version, released under a proprietary license, and in a community, version released under the Apache License.[30]

✓ **Python language:** Python is a programming language (e.g., C, C++, FORTRAN, Java...), developed in 1989. Its main characteristics are open source is free to use, source files are available and editable; Equipped with a very extensive core library and a large amount of libraries available for scientific computing, statistics, databases, visualization ... etc. Dynamic writing is done automatically during program execution, allowing great flexibility and speed in programming, but spurred by excessive memory consumption and performance loss, providing support for "incorporation of other languages" [31]. It was used in this work in order to extract the result curves.

✓ **C Language:** Is a compiled language (as opposed to interpreted languages). This means that a C program is written as a text file, called a source file. This file is obviously not executable by the microprocessor, it must be translated into machine language. This operation is performed by a program called a compiler [17]. It was used in this work in order to modify the files of the COOJA simulator according to the requirements of the study.

B. Hardware tools:

The characteristics of the computer in which the study is simulated are:

- ✓ Computer: ACER.
- ✓ Processor: intel (R) Celeron CPU N2840 @ 2.16GHz
- ✓ RAM:4.00 Go

V. Simulation Environment:

The main parameters of the simulation environment are summarized in Table (8). We study protocol simulation with the help of a network of 2, 15, 35 servers.

The nodes are first set randomly at a height of 50 x 100 meters. The knot moves at a rate of one meter per second. The movement of the model is determined by the maximum travel speed and the stopping time. Various scenarios will be simulated. It will take 3600 seconds to complete each simulation.

Here are the default values for the environment parameters below:

Settings	Value
Surface	50x100 m
Phase initialization	60 second
Time simulation	3600s
Randomseed (Random speed)	123,456
Mote startup delay	5 S
Number of servers	2,15, and 35
Congestion mechanisms	CoAP
Max retransmissions	4
Radio band	3.9 GHz
Physical	IEEE 802.15.4 PHY
Distances between nodes	10M
Transport and network	UDP,RPL

Table 9: Simulation environment

VI. Simulation scenario: Smart City

To evaluate the performance of the protocol, we create a smart City that uses CoAP to communicate with restricted devices:

A smart city is the actual “smart planet” approach applied in a particular area, i.e. the application of information and communication technology to sense, analyze and integrate information into managing cities to meet various needs, including daily livelihoods, environmental protection, public safety, industrial and commercial activities, and also improving Transport and social services. [11] The main objective of a smart city is to improve

city functions and promote economic growth while improving the quality of life for citizens using smart technologies and data analysis.

Image (21) shows components of smart City



Image21: Smart City components

Adoption of the IoT model in a smart city scenario is very attractive to public administrations, which may promote IoT adoption on a larger scale.

The image shows an example of smart cities: The Internet of Things in the city of PADOVA, Italy, realized the smart city project, where the control program was developed for the IoT nodes, which consists of a system for monitoring public street lighting, measurements of light intensity at each location, measurements of carbon dioxide level, air temperature, Humidity, vibration and noise. This system is a simple implementation of the IoT concept, it still includes a number of different devices and link layer technologies, and thus represents the most critical issues that need to be considered when designing urban IoT. [73]

VII.CoAP simulation using COOJA:

We'll use COOJA to run the CoAP protocol and establish a connection between the border router and other nodes in this part. The following are the steps:

➤ **Step 1:** we open VMware window:

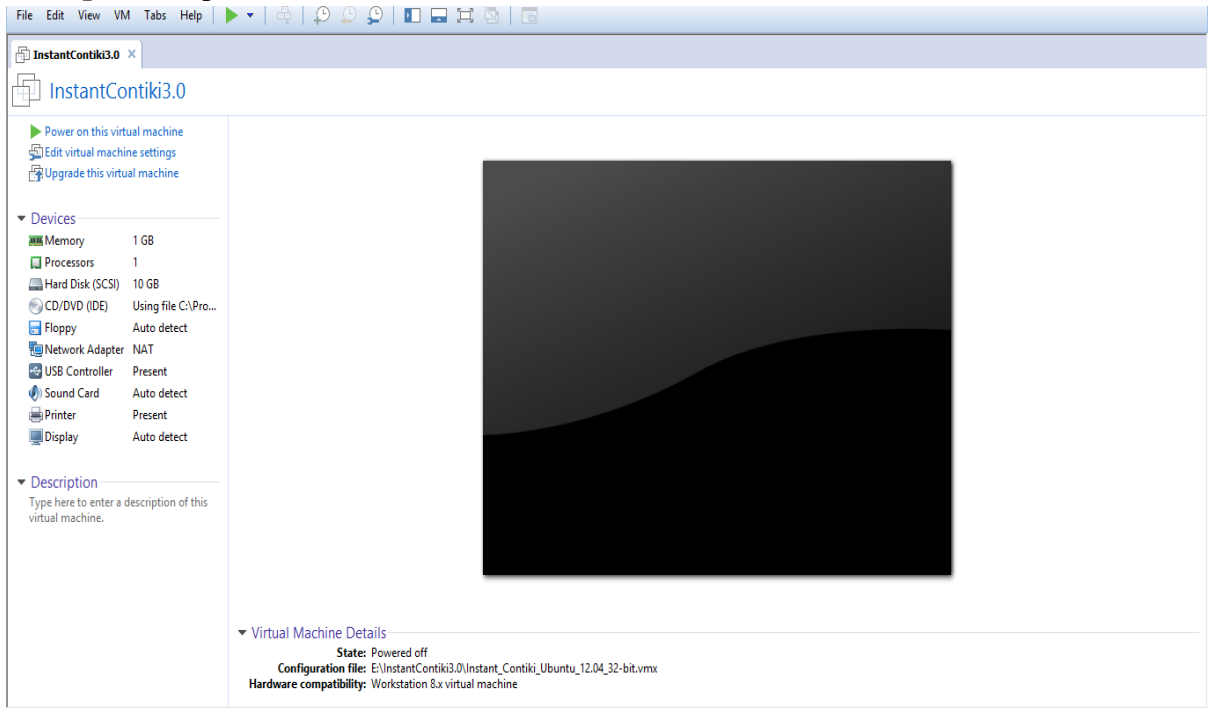


Image 22: VMware window

➤ **Step 2:** we open Contiki OS:

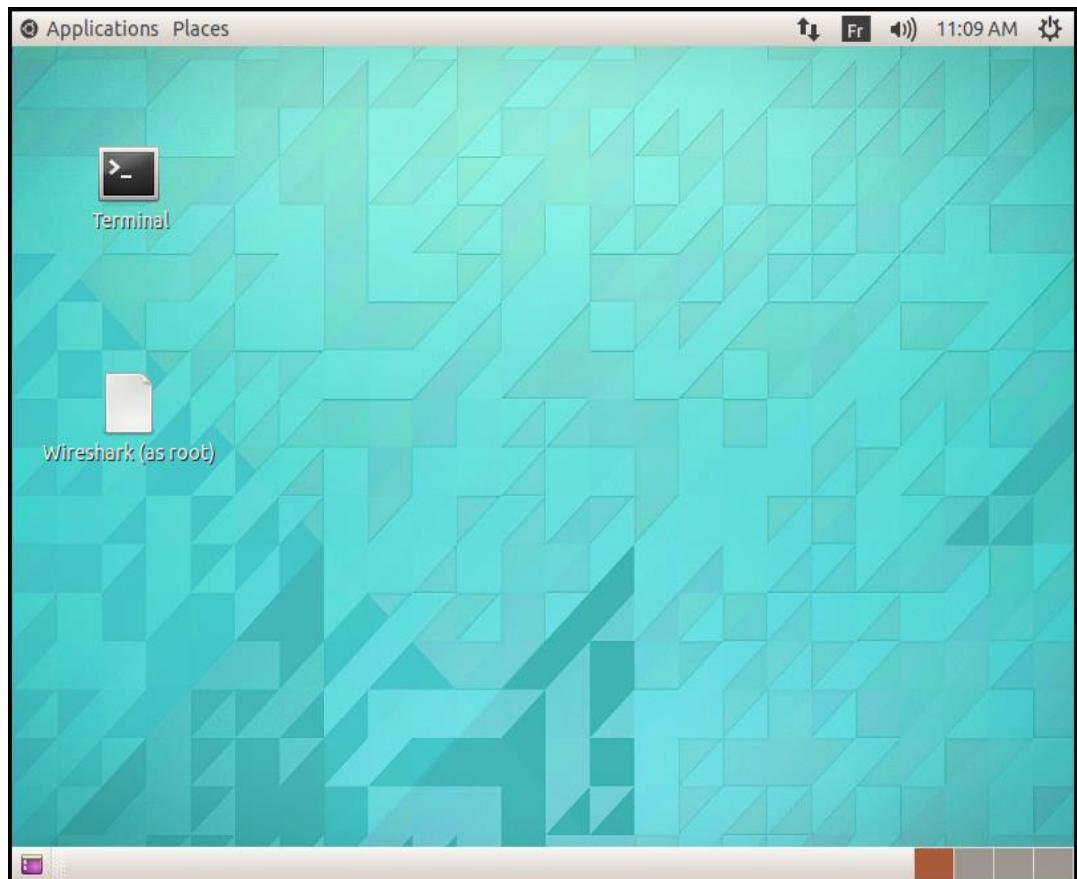


Image 23: Contiki OS window

- **Step 3:**we open Cooja simulator then open new simulation

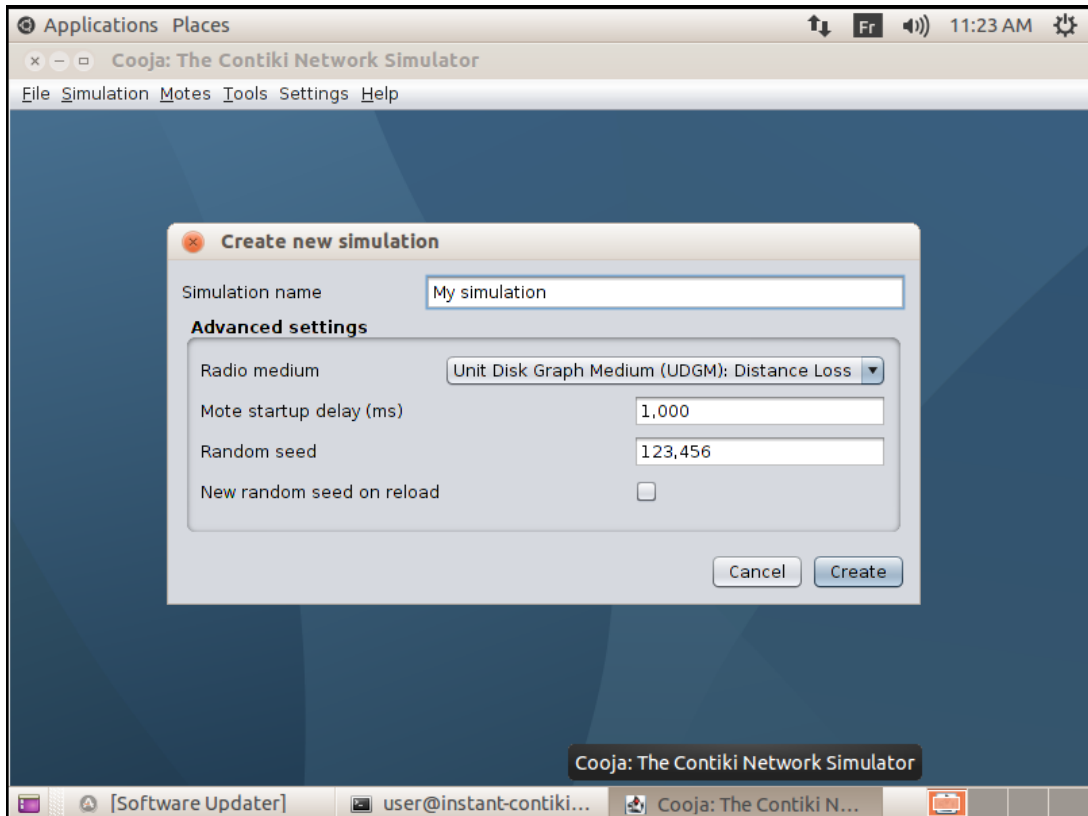


Image 24: Cooja simulator

- **Step 4:** We open a new simulation, Open notes menu >> add notes >> create new notes type>>sky, three files are necessary to run CoAP applications. In order to create the notes :border-router.c,er-example-server.c, er-example-client.c.

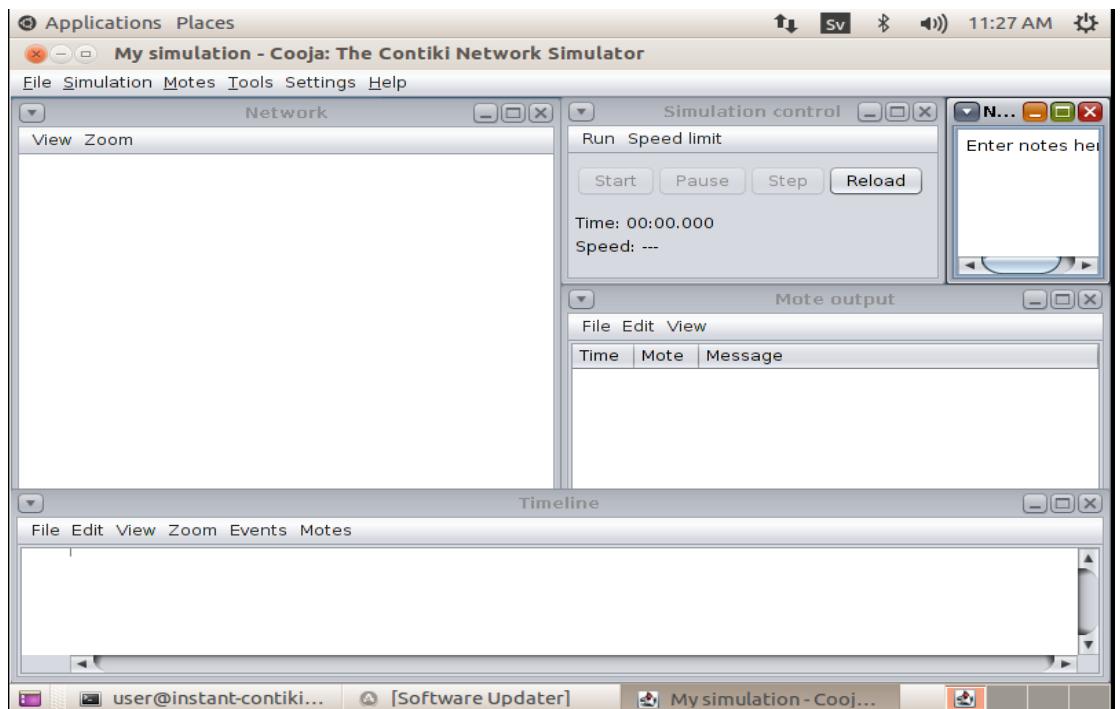


Image 25: COOJA window

- **Step 5:** To create border router motes: home/user/contiki/examples/ipv6/rpl-border-router/border-router.c, Choose the file in location >> compile >> create >> Add motes.

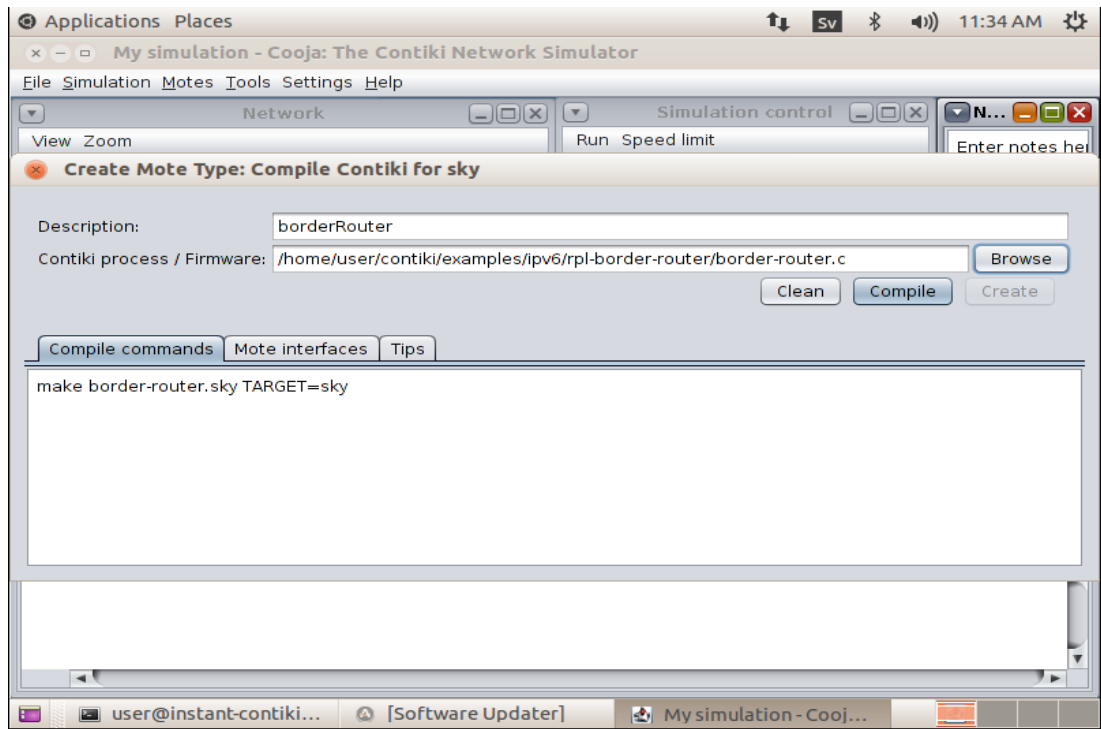


Image 26: Border Router Mote

- **Step 6:** To create server motes : /home/user/contiki/examples/er-rest-example/er-example-server.c, choose a file in location >> compile >> create >> choose server >> Add motes

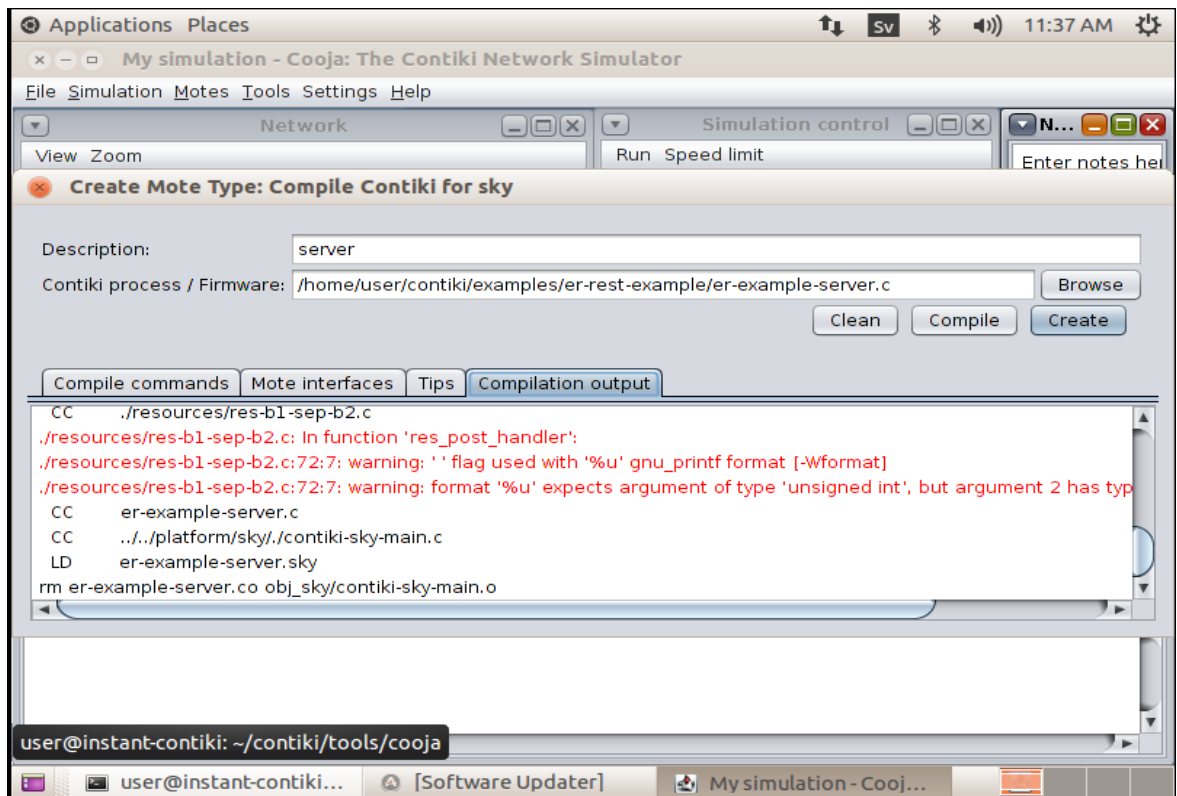


Image27: CoAP server motes

- **Step 7:** To create client notes : /home/user/contiki/examples/er-rest-example/er-example-client.c, choose a file in location >> compile >> create >> choose server >> Add notes

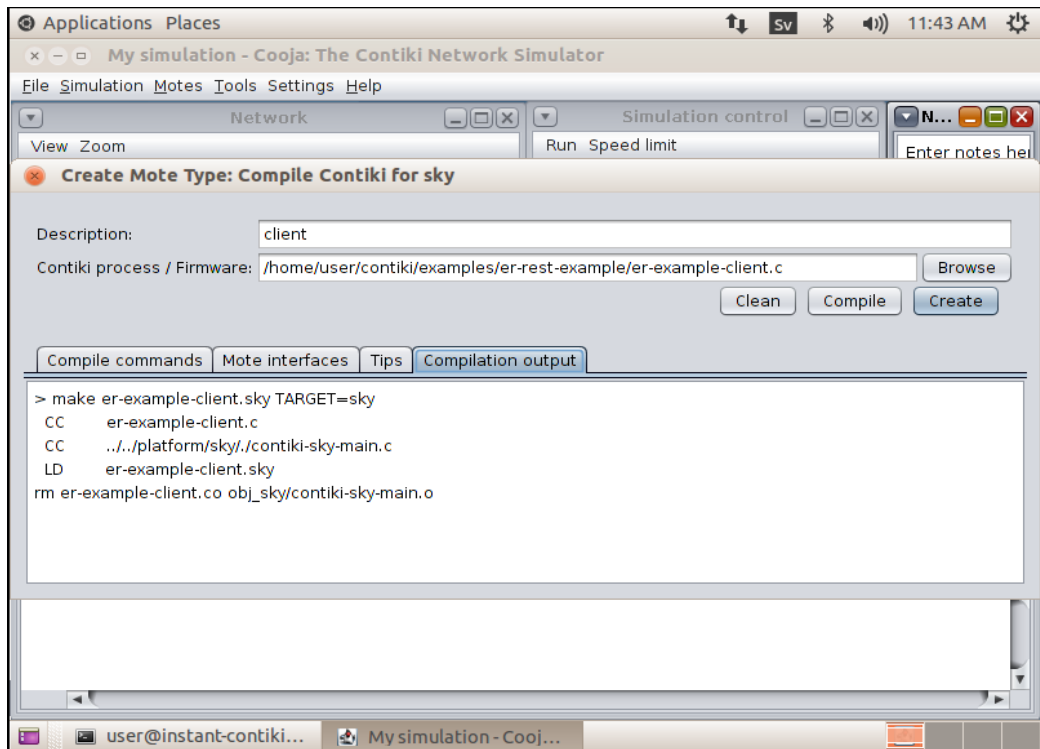


Image 28: CoAP client mote

- **Step 8:** in order to make a connection between the border router and other nodes, we must enable a bridge by Right Click Border Router Node -> Mote tools for Sky 1 -> Serial Socket (SERVER) -> start.

On the other hand, we open a new terminal, and in the following path /home/contiki/examples/ipv6/rpl-border-router/ we do the command: make connect-router-COOJA.

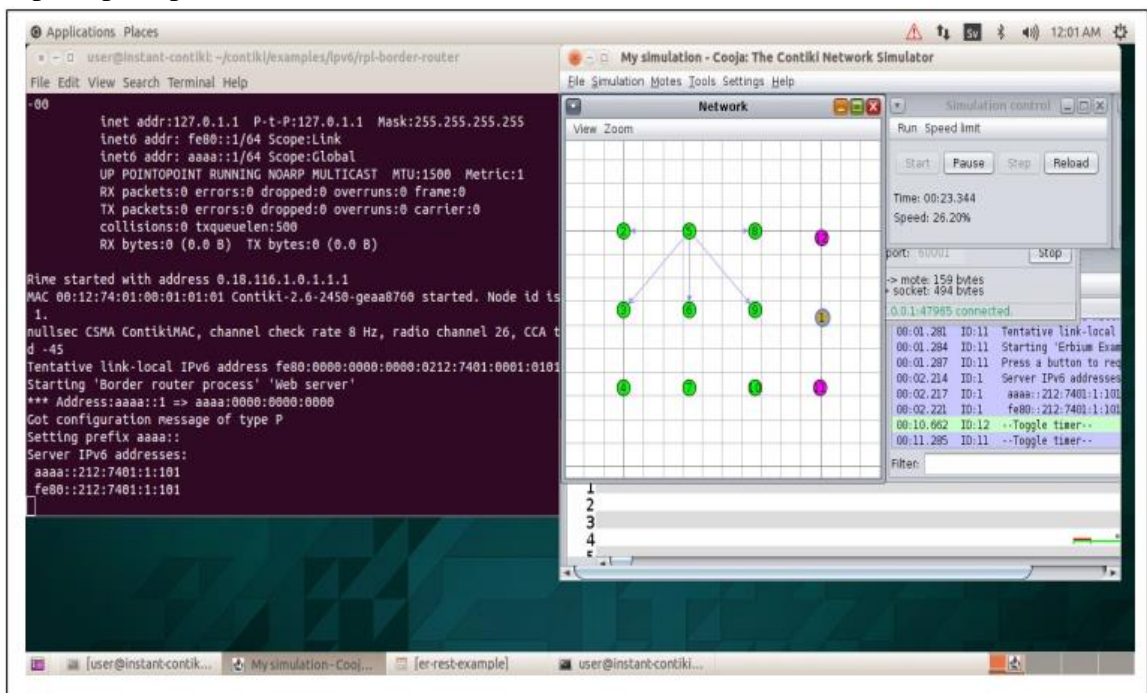


Image 29: Border router and CoAP servers

✓ **Step9:** In order to get the results in form of curves , we used PyCharm program ,which works as follows: after the simulation we copied the obtained code into file (.txt) ,then opened it directly with PyCharm and executed it.

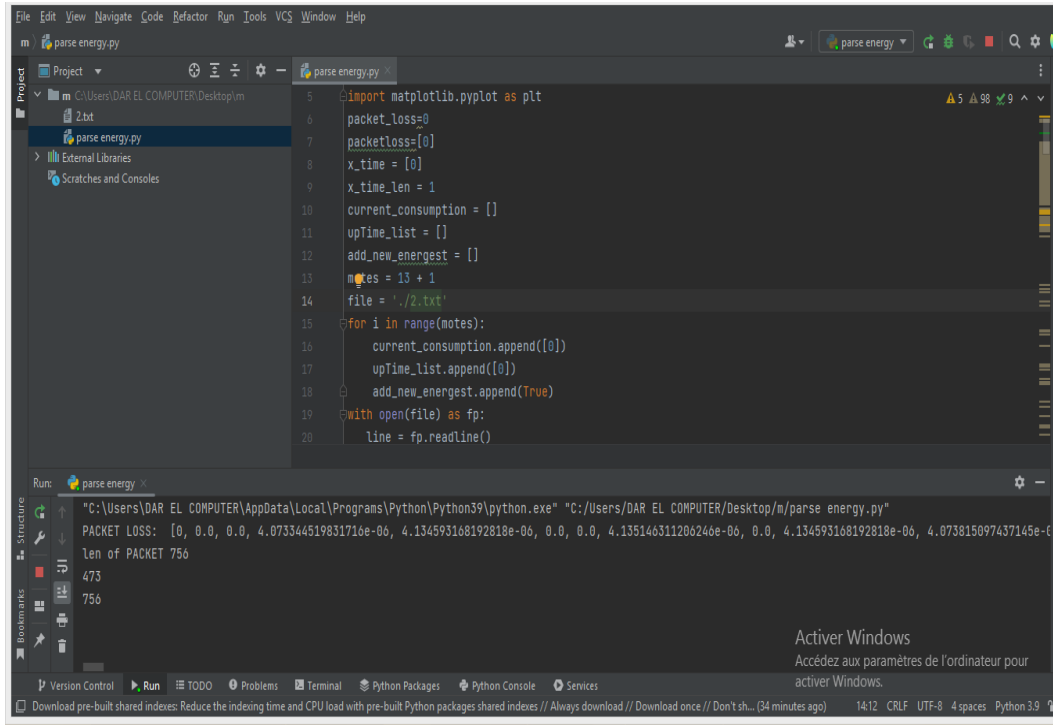


Image 30: PyCharm interface

VIII. Performances evaluation experiments:

In order to evaluate CoAP protocol performances, we use in our study one parameter (number of servers) where we study each parameter with energy consumption and packet loss. We did this by simulating the protocol and modifying the files provided in the system by adding some codes.

In each time we will change number of servers and topology take results and transform them in curves in order to analyze the protocol.

❖ Experiment 1: Number of servers with Random Topology

In this experiment, we will study the energy consumed and the loss of packet according to the number of servers in random topology. The first is simulated by 2, then 15, 35 servers with 10 clients, showing 47, 48, 49 simulation formats according to the server number.

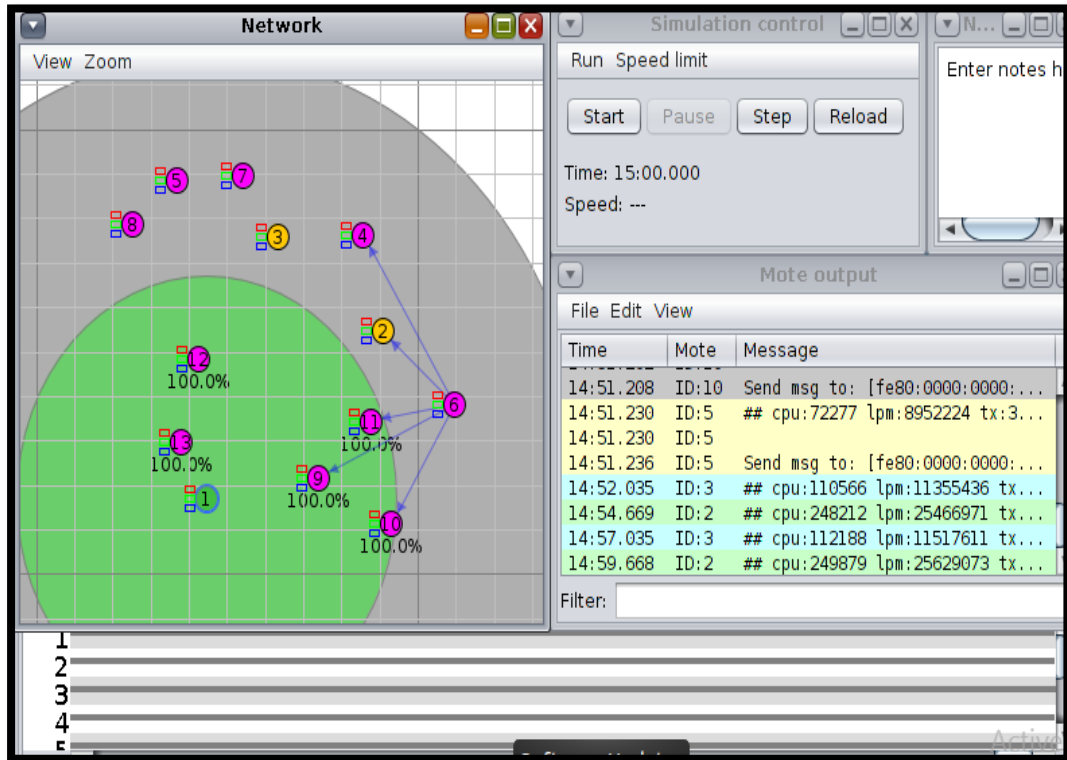


Image 31: Simulation of 2 Servers in Random Topology

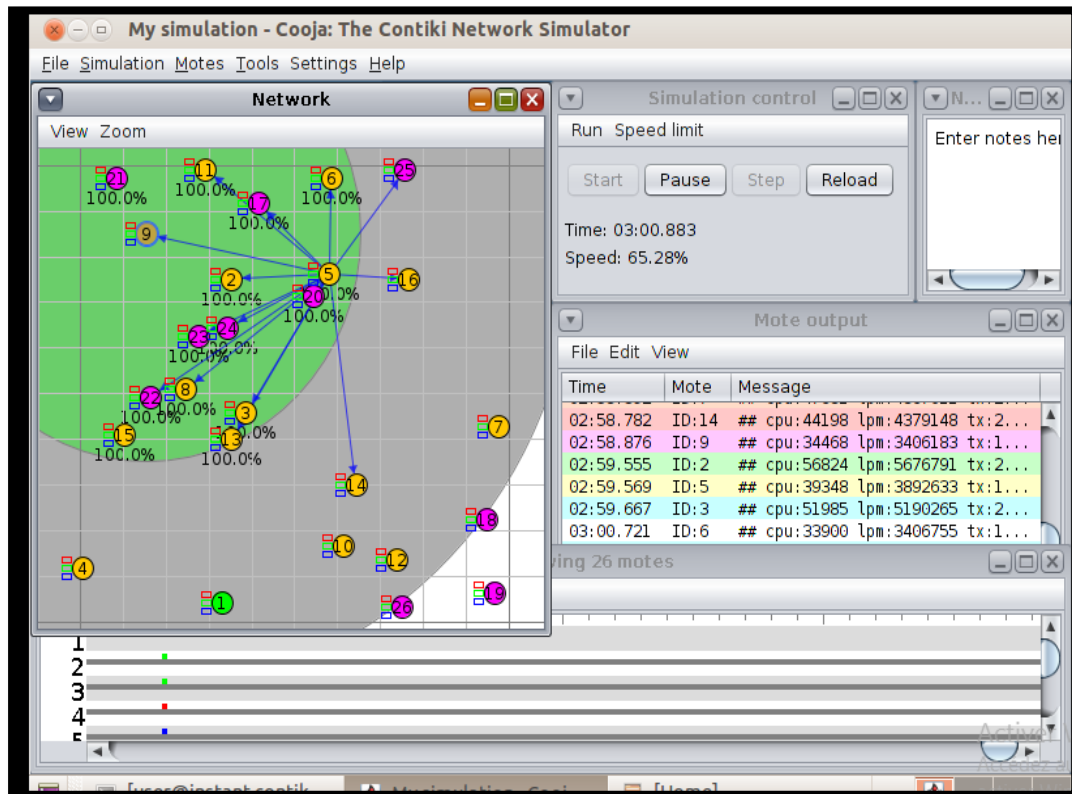


Image32: Simulation of 15 Servers in Random Topology

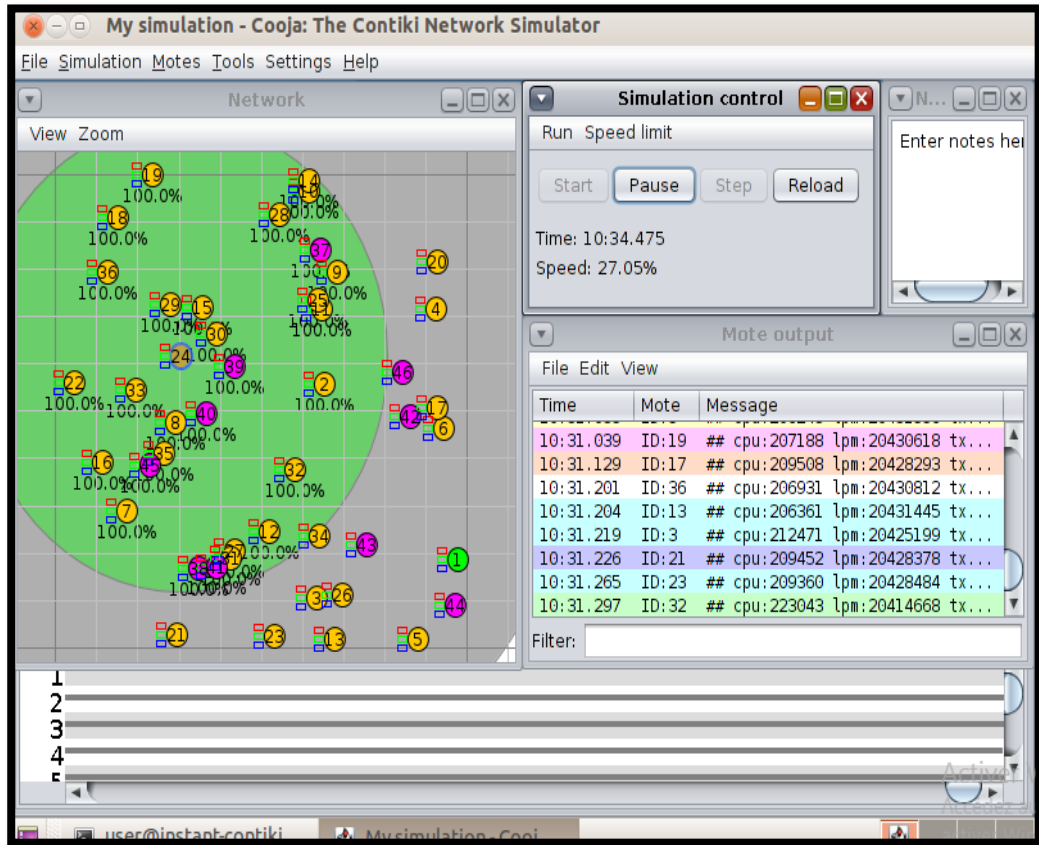


Image 33: Simulation of 35 Servers in Random Topology

❖ **Experiment 2: Number of servers with Linear Topology**

In this experiment, we will study the energy consumed and the loss of packet according to the number of servers in Linear topology with 10 client. The first is simulated by 2, then 15, 35 servers, showing 50, 51, 52 simulation formats according to the server number.

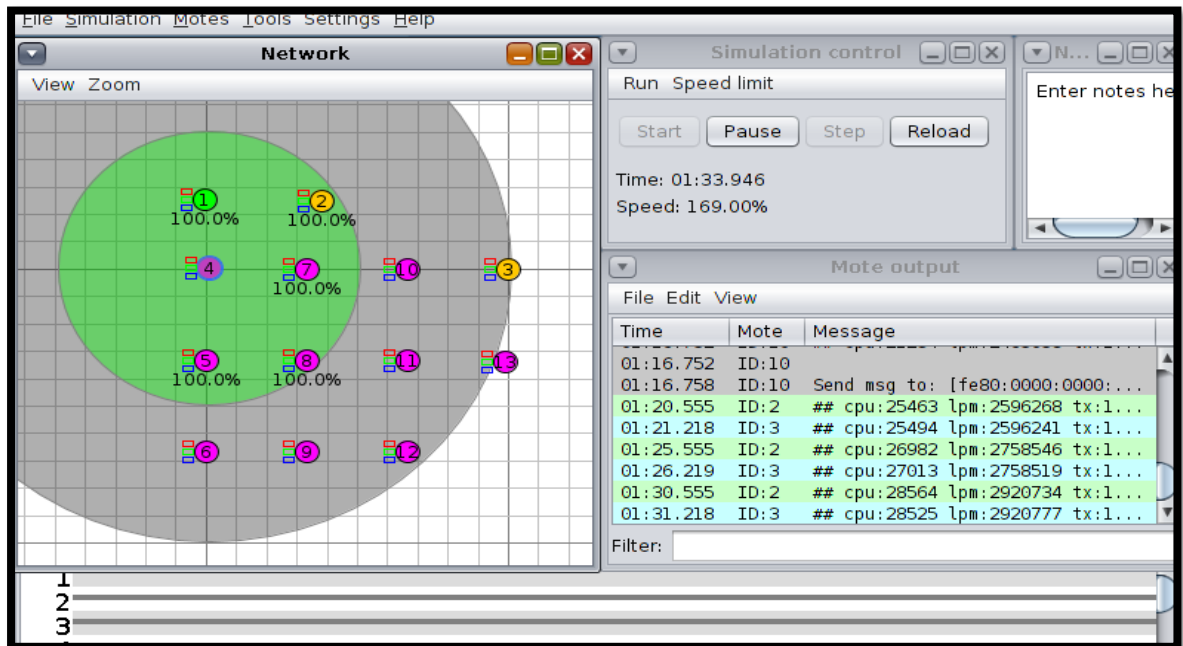


Image 34: Simulation of 2 Servers in Linear Topology

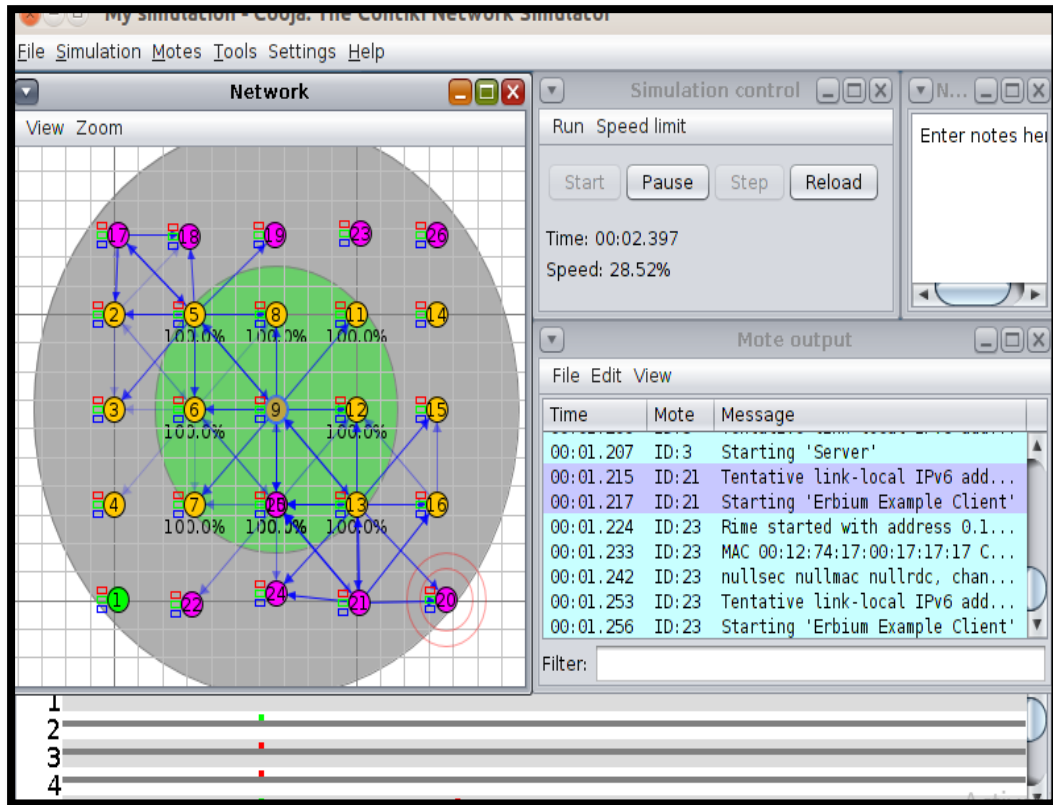


Image35: Simulation of 15 Servers in Linear Topology

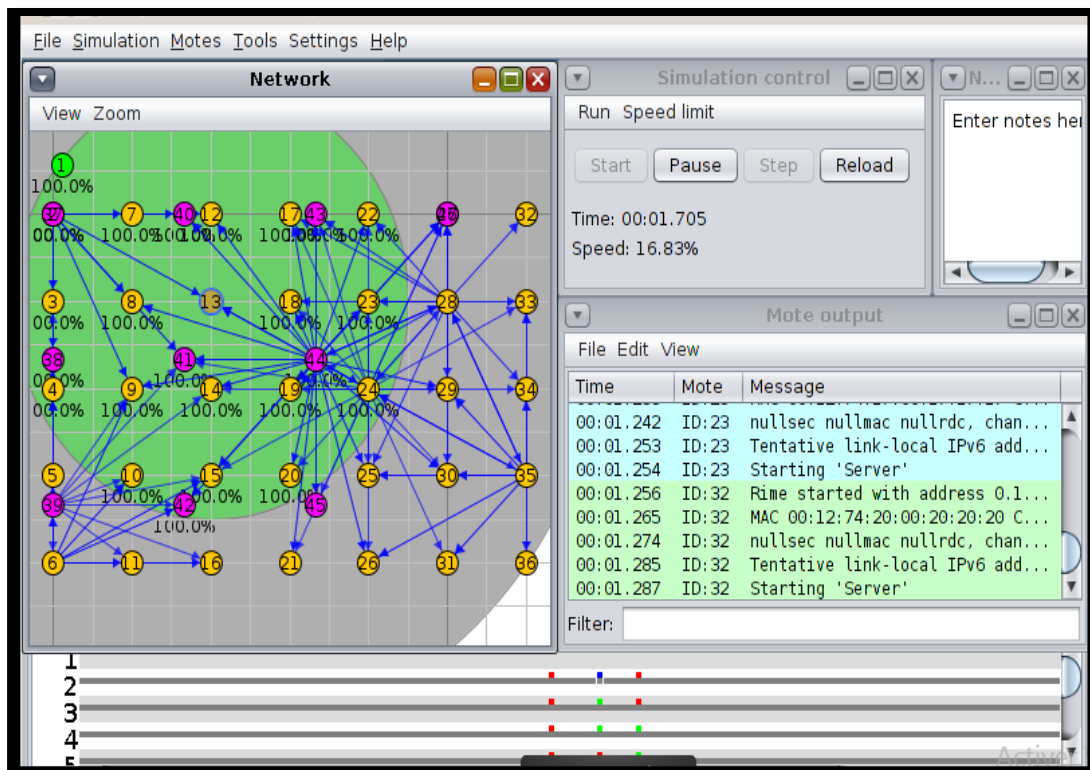
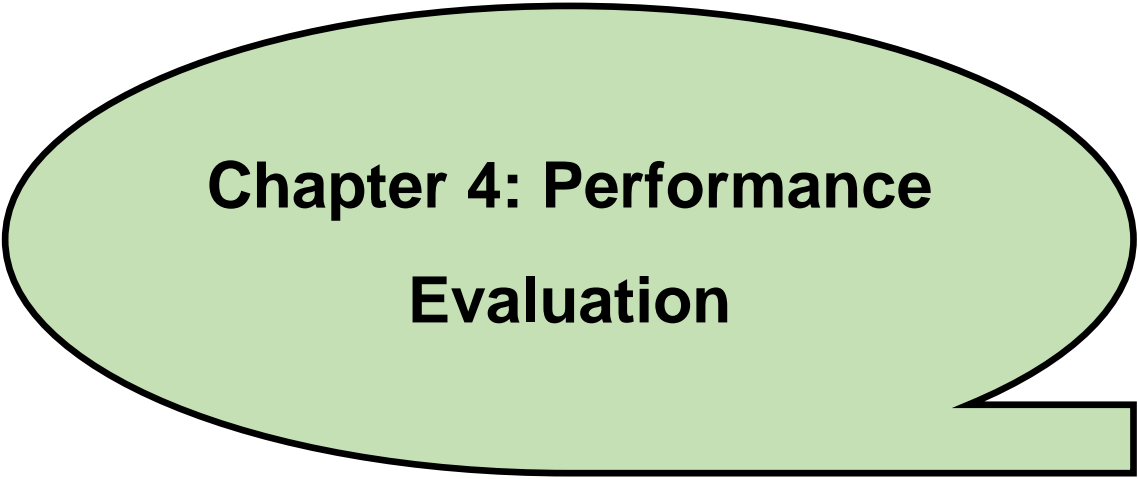


Image 36: Simulation of 35 Servers in Linear Topology

IX. Conclusion:

For the purpose of simulation of the CoAP protocol, we have presented a presentation of the protocol simulation in this chapter. We begin by defining the validation methods, and then we moved on to present the programs used, as well as the simulation environment. Finally, we provide several simulations by increasing the number of servers in several network topologies in order to calculate the energy consumption and the packets loss in CoAP protocol according to the different proposed topologies.

In the following chapter, we will give and analyze the results of the experiments to evaluate the protocol performances.



**Chapter 4: Performance
Evaluation**

I.Introduction:

The selection of a model, the evaluation using a simulation consists of and the interpretation of the measurements acquired are all part of the simulation evaluation of a system's performance. For the study of architectures and protocols in diverse network contexts, a significant variety of simulation models have been built (number of nodes, mobility, etc.). They have been commonly employed in routing protocol evaluations. [33].

In this chapter, we will discuss the results obtained through simulations in the previous part, and we'll conclude that the CoAP protocol is effective for the Internet of Things based on them.

II.Standards of Evaluation:

To evaluate the performance of the protocol, we study some Measures, including [36, 38, 40]:

1. Network Delay:

This performance metric is used to measure the average end-to-end delay of data packet transmission. The end-to-end delay implies the average time taken between a packet initially sent by the source, and the time for successfully receiving the message at the destination. measuring this delay takes into account the queuing and the propagation delay of the packets. It is the sum of 2 * max latency and the processing delay.

2. Network Throughput:

The end-to-end network throughput measures the number of packets per second delivered at the destination. It is considered here as an external measure of the effectiveness of a protocol. is calculated as (in Kbps) = (No of successful CoAP request/response pairs * (length of request + length of response in bits)) / total time of simulation

3. Packet Delivered Successfully:

The total number of packets delivered at the destinations versus the total number of packets sent from the source.

4. Latency:

The average message latency is defined as the average amount of time between the start of distributing data and its arrival at a node interested in receiving the data. hence the latency measures time performance for the individual message.

5. Energy Consumption:

Is the sum of used energy of all the nodes in the network, where the used energy of a node is the sum of the energy used for communication, including transmitting (pt), receiving (pr), and idling (pi). Assuming each transmission consumes an energy unit, the total energy consumption is equivalent to the total number of packets sent in the network.

Power Consumption= (Transmit/19.5 mA + Listen /21.5 mA +CPU power/1.8 mA +LPM/0.0545 mA)/3v/ (32768).

6. Network Lifetime:

It is considered as the time until the message loss rate is above a given threshold. the more complete definition for the lifetime of the network is “time to network partition” network partition occurs when there is a cut-set in the network. it will be introduced as a new metric, which will use energy variance:

$$\text{network lifetime} = e - (u + \sigma), \text{ where } u$$

e is the total initial energy at each node (full battery charge),

ui is the average used energy,

n is the total number of nodes in the network,

σ is expressed as

$$\sigma^2 = \frac{(u_i - u)^2}{n}$$

All these metrics are calculated using their cumulative average values, that is, at time t, the performance value is the average from 0 to t (seconds).

7. Packet Generation Rate:

It is the number of packets that the sensor node transmits in one time period which is usually one second.

III.Performance evaluation method:

In our work we have choose to evaluate the CoAP protocol by evaluating two metrics which are energy consumption and number of packet loss in 2 topologies: random and linear topology.

Through the changes mentioned in Chapter Three regarding Contiki files, we can see the packets loss and energy consumed by the servers through the mote-output window in Cooja.

To evaluate the protocol performance, we have done the following experiments:

❖ **Experiment1 results: Number of Server with Random Topology**

Figure 37: shows the energy consumption in relation to the Number of servers.

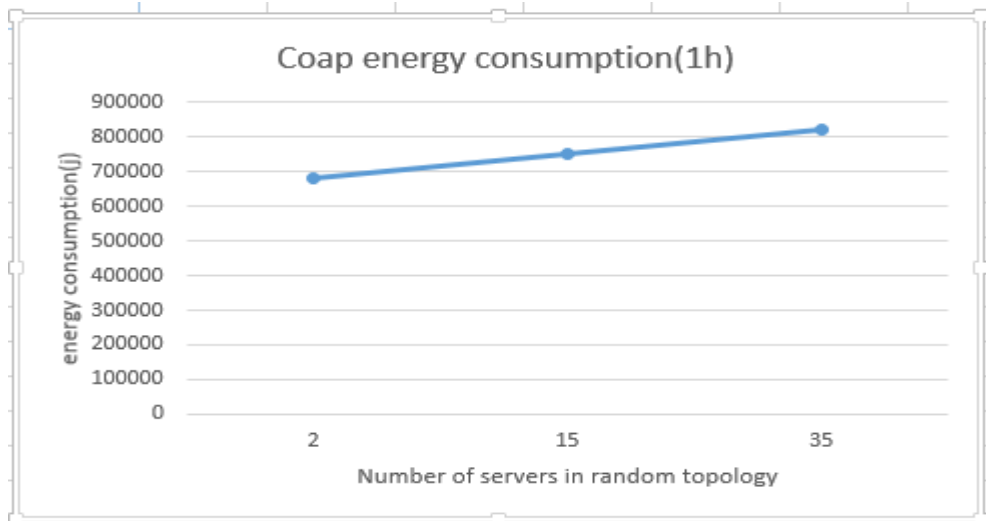


Image 37: Curves of energy consumption (j) against the Number of COAP servers in 1 h

From figures (37), we note the value of power consumption when consuming 2 servers less than power consumption when consuming 15 and 35 servers.

✓ Figure 38: shows the energy consumption in relation to the Number of servers.

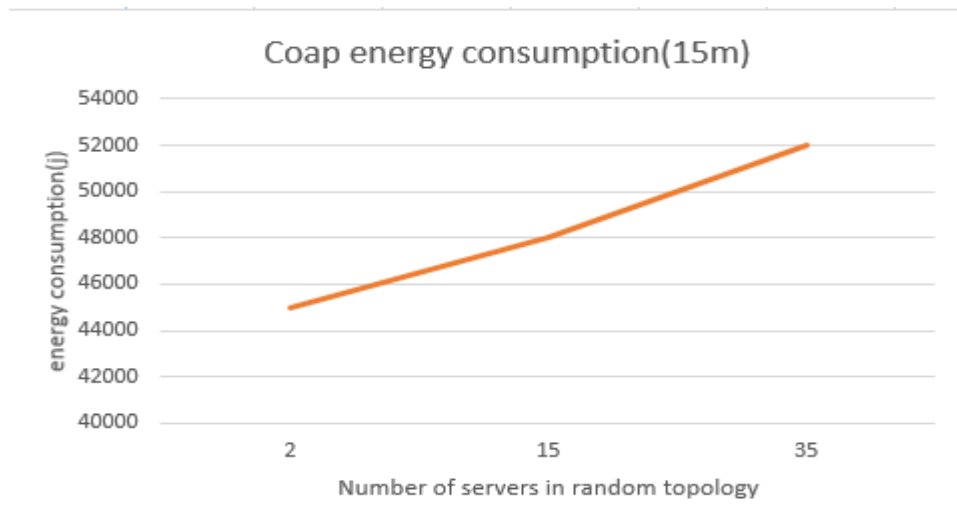


Image 38: Curves of energy consumption (j) against the Number of COAP servers in 15m

From figures (38), we note the value of power consumption when consuming 2 servers less than power consumption when consuming 15 and 35 servers.

✓ Figure 39: shows the packets loss in relation to the number of servers.

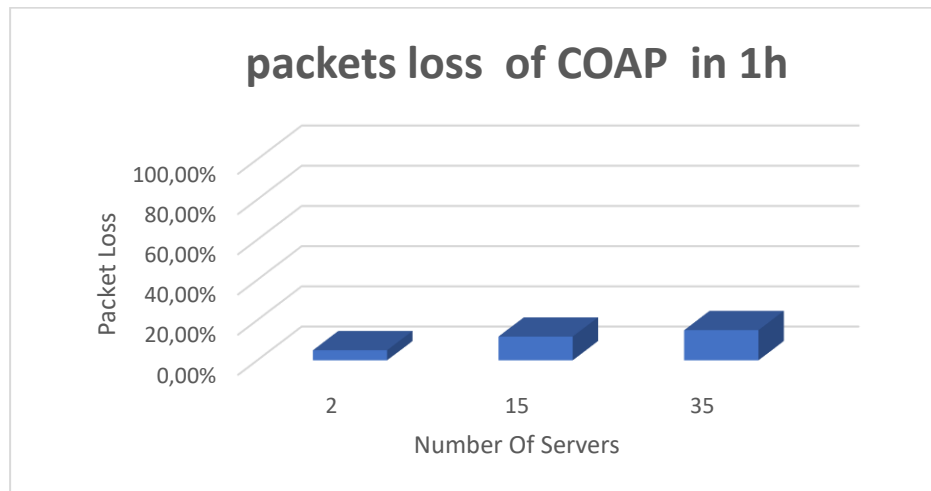


Image 39: Curves of packets loss against the Number of COAP servers in 1h

From figures (39), we see a large loss of packets at the beginning of the simulation and then it decreases gradually, and the increase in the number of servers leads to a greater loss of packets but with low loss rate.

✓ Figure 40: shows the packets loss of 2,15 and 35 servers in relation to the time(15 m).

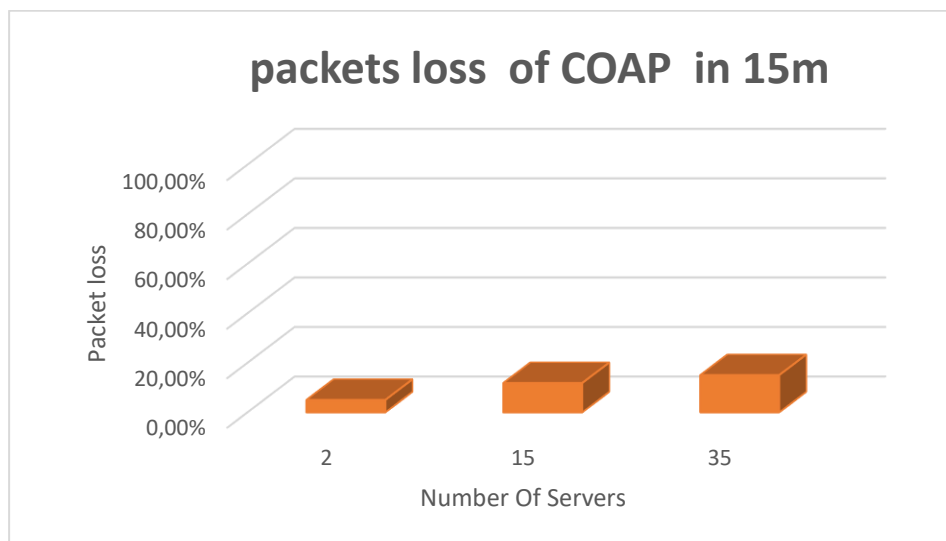


Image 40: Curves of packets loss against the Number of COAP servers in 15m

From figures (40), we see a large loss of packets at the beginning of the simulation and then it decreases gradually, and the increase in the number of servers leads to a greater loss of packets but with low loss rate.

❖ **Results Simulation Experiment2: Number of Server in Linear Topology**

✓ Figure 41: shows the energy consumption of 2,15 and 35 servers in relation to the time(1h).

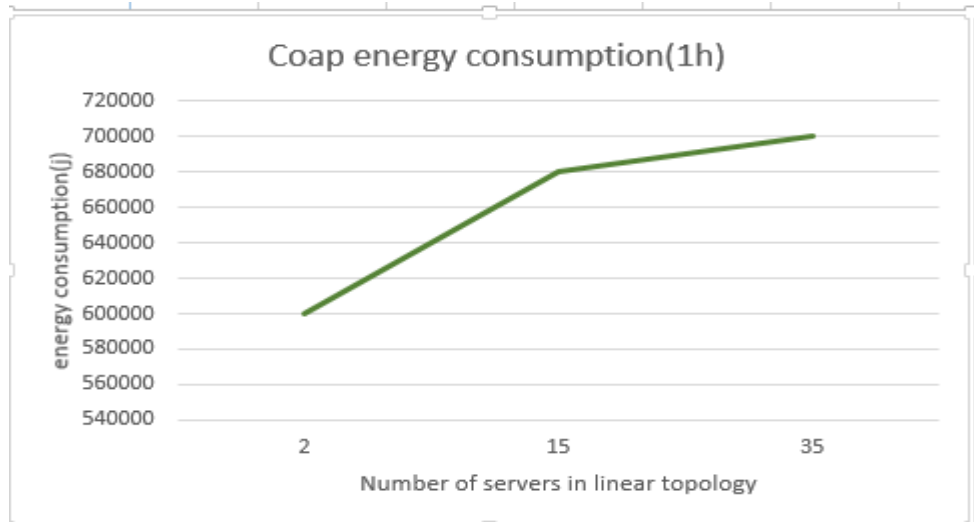


Image 41: Curves of energy consumption (j) against the Number of COAP servers in 1 h

From figures (41), we note the value of power consumption when consuming 2 servers less than power consumption when consuming 15 and 35 servers.

✓ Figure 42: shows the energy consumption in relation to the number of servers.

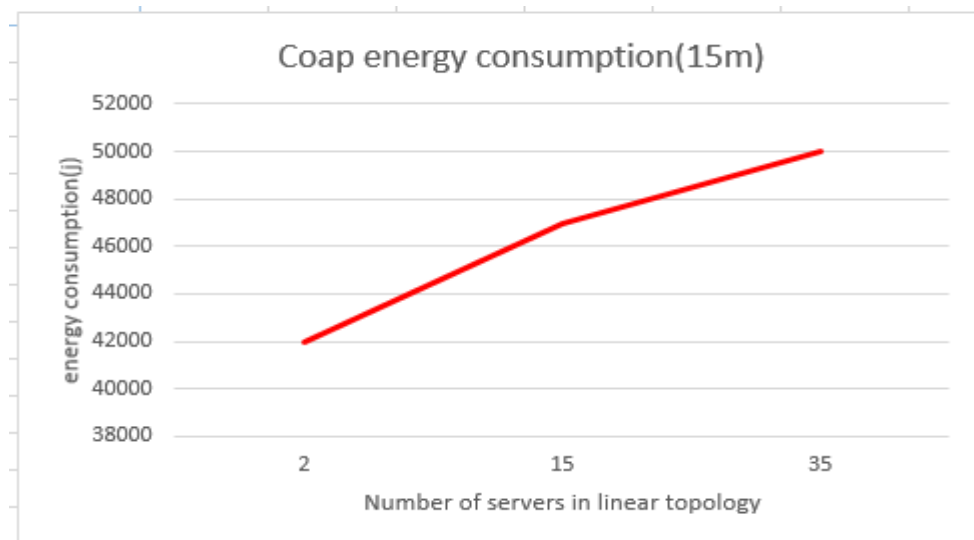


Image 42: Curves of energy consumption(j) against the Number of COAP servers in 15m

From figures (42), we note the value of power consumption when consuming 2 servers less than power consumption when consuming 15 and 35 servers.

✓ Figure 43: shows the packets loss of 2,15 and 35 servers in relation to the time(1 h).

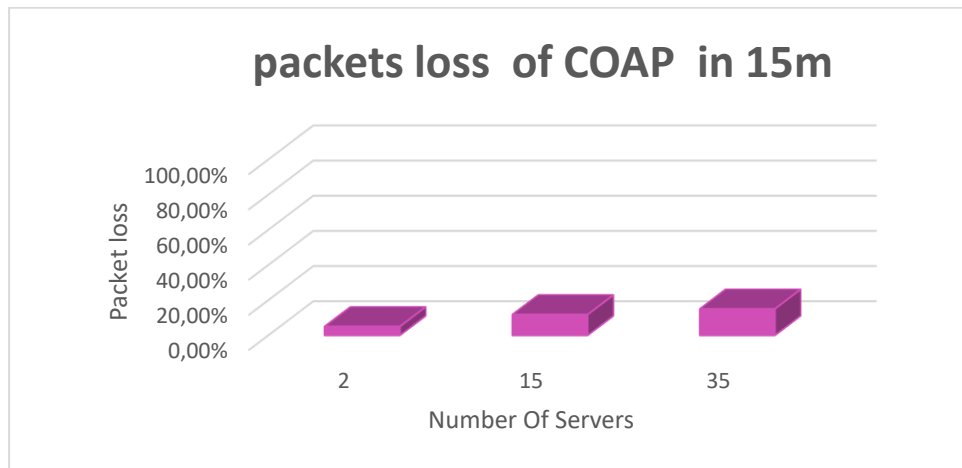


Image 43: Curves of packets loss against the Number of COAP servers in 1h

From figures (43), we see a large loss of packets at the beginning of the simulation and then it decreases gradually, and the increase of the number of servers leads to a greater loss of packets but with low loss rate.

✓ Figure 44: shows the packets loss of 2,15 and 35 servers in relation to the time(15 m).

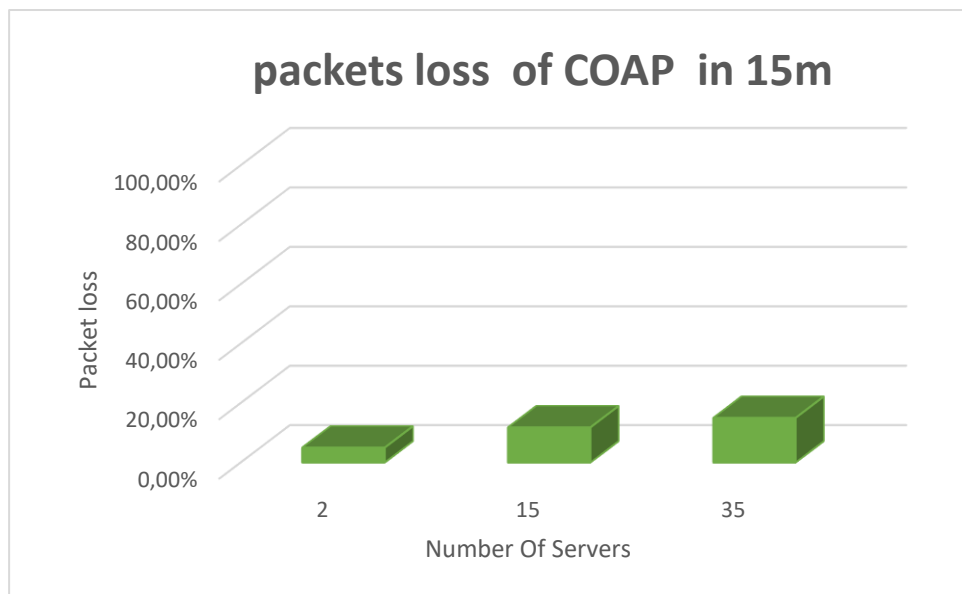


Image 44: Curves of packets loss against the Number of COAP servers in 15m

From figures (44), we see a large loss of packets at the beginning of the simulation and then it decreases gradually, and the increase in the number of servers leads to a greater loss of packets but with low less rate.

✓ **Analyze:**

When studying packet loss, as well as power consumption, with a gradual increase in the number of servers:

- At the beginning of the simulation, when all servers are listed at once, we notice a large loss in the number of lost packets because they are not sorted yet, and over time the loss gradually decreases due to their reading one by one and receiving them well.
- Increasing the number of servers leads to an increase in energy consumption because each server needs several operations to analyze it, read it, and so on. All of these processes consume abundant energy for good operation and its guarantee.
- When the number of servers increases, the radio traffic increases due to the difficulty of reaching the border router, which explains the presence of packet loss, that is, the greater the number of servers, the greater the packet loss.
- The shape of the topology does not affect the results, because whatever the topology is, the same number of servers and clients and the same processes remain.
- The process of sending and receiving parcels in the vicinity of the simulation increases energy consumption. It is not easy to authenticate between a client and servers, as well as ensuring the transfer of information and not losing any of them. All of this requires abundant time and energy.

IV. Evaluation results:

After simulations performed with COOJA and analyzing the results shown in the form of a graphical curve, the results were arrived at for performance evaluation:

- Packet loss is affected when the number of servers increases, but with a small interval.
- The increasing number of servers affects energy consumption too.

Finally, it was concluded that CoAP achieves reliability and scalability. One of its weaknesses is the high power consumption.

CoAP achieves low packet loss rate, and is a good protocol for communication with IoT applications.

V.Conclusion:

In this chapter, we present an illustrative study to evaluate the performance of the protocol through the measurements extracted from our experiments.

We analyze from the obtained results that with CoAP protocol when we increase the number of servers there are an increase of the energy consumption. There is also an increase of the ration of packets loss spatially in the earlier time of the simulation then it became stable after a few moment.

Conclusion General:

The Internet of Things has become the most important technology of our time for its many benefits, for facilitating communication and dealings in various sectors, as well as for resolving all issues related to modern automated industries around the world.

Since protocols are the most important factor in communications, we focused on them in this work, and due to their multiplicity, it is difficult to choose the optimal protocol for use in the Internet of Things. After the theoretical study of the Internet of Things, we found that the COAP protocol is the most widely used, so we evaluate its performance through simulations in the COOJA network simulator under verity of conditions to see how effective this protocol is.

Among the programs and tools used in COOJA Simulator based on Contiki-os and PyCharm, in order to establish a reliable connection between clients and servers, several criteria have been chosen to study and evaluate the protocol: power consumption and lost packets to see how effective they are with the characteristics of the Internet of Things. This work was carried out in four stages:

- The first phase includes a study of the Internet of Things, including its benefits, applications, characteristics, and protocols.
- Phase 2: A theoretical study of the COAP protocol as well as with the characteristics of the Internet of Things, in addition to a study of the HTTP protocol due to their similarities.
- Phase 3: presentation of how to work with COOJA and PyCharm to get results with a define experiment.
- Phase 4: Effectiveness, study of results and evaluation of the protocol.

Finally, it was concluded that CoAP achieves reliability and scalability, and one of its weaknesses is the high power consumption.

CoAP achieves little packet loss over simulation time after increment in it, and is a good protocol for communication with IoT applications.

As a future work , we think about some suggestions that can be added , including:

- Evaluate COAP's performance with other features that we haven't used as congestion and thought put.
- Studying the COAP protocol using another mathematical analysis method.
- Improve the COAP protocol to become the basic protocol in the Internet of Things application layer

❖References:

- [1] Wu, M., Lu, T. J., Ling, F. Y., Sun, J., & Du, H. Y. (2010, August). Research on the architecture of Internet of Things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)* (Vol. 5, pp. V5-484). IEEE.
- [2] Gokhale, P., Bhat, O., & Bhat, S. (2018). Introduction to IOT. *International Advanced Research Journal in Science, Engineering and Technology*, 5(1), 41-44.
- [3] Alamri, M. H. (2017). Securing the Constrained Application Protocol (CoAP) for the Internet of Things (IoT).
- [4] Salman, T., & Jain, R. (2015). Networking protocols and standards for internet of things. *Internet of things and data analytics handbook*, 7, 14-18.
- [7] Patel, K. K., & Patel, S. M. (2016). Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. *International journal of engineering science and computing*, 6(5).
- [8] Lao, L., Li, Z., Hou, S., Xiao, B., Guo, S., & Yang, Y. (2020). A survey of IoT applications in blockchain systems: Architecture, consensus, and traffic modeling. *ACM Computing Surveys (CSUR)*, 53(1), 1-32.
- [9] Challal, Y. (2012). *Sécurité de l'Internet des Objets: vers une approche cognitive et systémique* (Doctoral dissertation, Université de Technologie de Compiègne).
- [10] Vilamovska, A. M., Hattziandreu, E., Schindler, R., Van Oranje, C., De Vries, H., & Krapelse, J. (2009). Rfid application in healthcare—scoping and identifying areas for rfid deployment in healthcare delivery. *RAND Europe, February*, 26.
- [11] Su, K., Li, J., & Fu, H. (2011, September). Smart city and the applications. In *2011 international conference on electronics, communications and control (ICECC)* (pp. 1028-1031). IEEE.
- [12] Kaur, S., & Singh, I. (2016). A survey report on Internet of Things applications. *International Journal of Computer Science Trends and Technology*, 4(2), 330-335.
- [13] Malika, H., & Nabila, T. (2019). *Internet des objets dans le domaine de l'agriculture de demain* (Doctoral dissertation, Université Mouloud Mammeri).
- [14] Ouadah, A. (2019). *Modélisation et vérification formelle d'un protocole CoAP pour l'internet des objets* (Doctoral dissertation, UNIVERSITE MOHAMED BOUDIAF-M'SILA FACULTE DES MATHÉMATIQUES ET DE L'INFORMATIQUE DEPARTEMENT D'INFORMATIQUE-Option: Système d'information et génie logiciel Spécialité: Informatique Décisionnel et Optimisation).
- [15] Karagiannis, V., Chatzimisios, P., Vazquez-Gallego, F., & Alonso-Zarate, J. (2015). A survey on application layer protocols for the internet of things. *Transaction on IoT and Cloud computing*, 3(1), 11-17.
- [17] Canteaut, A. (2003). Programmation en langage C. *INRIA-projet CODES*.
- [18] Locke, D. (2010). Mq telemetry transport (mqtt) v3. 1 protocol specification. IBM developerWorks Technical Library (2010).
- [19] Saint-Andre, P. (2011). *Extensible messaging and presence protocol (XMPP): Core* (No. rfc6120).
- [20] Standard, O. A. S. I. S. (2012). Oasis advanced message queuing protocol (amqp) version 1.0. *International Journal of Aerospace Engineering Hindawi www. hindawi. com, 2018*.

- [22] Verma, S., & Rastogi, M. A. (2020). Iot application layer protocols: A survey. *Journal of Xi'an University of Architecture & Technology*, 12.
- [31] "Introduction A La Programmation En Langage Python », Université Paris-Sud ,Méthodologie Licence Mpi S2 - Année 2015-2016
- [33] Moussa, M. (2014). *Vérification et configuration automatiques de pare-feux par Model Checking et synthèse de contrôleur* (Doctoral dissertation, École Polytechnique de Montréal).
- [34] Chen, X. (2014). Constrained application protocol for internet of things. URL: <https://www.cse.wustl.edu/~jain/cse574-14/ftp/coap>.
- [35] Seoane, V., Almenares, F., Campo, C., & Garcia-Rubio, C. (2020, November). Performance evaluation of the CoAP protocol with security support for IoT environments. In *Proceedings of the 17th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks* (pp. 41-48).
- [36] Halder, M., Sheikh, M., Rahman, M., & Rahman, M. (2018). Performance analysis of CoAP, 6LoWPAN and RPL routing protocols of IoT using COOJA simulator. *Int. J. Sci. Eng. Res*, 9(6), 1670-1677..
- [38] Raj, S., & Rajesh, R. (2022). Routing Protocol for Low Power and Lossy Network Using Energy Efficient Priority Based Routing. *Wireless Personal Communications*, 123(2), 1379-1394.
- [40] Fournaris, A. P., Giannoulis, S., & Koulamas, C. (2019, June). Evaluating CoAP end to end security for constrained wireless sensor networks. In *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)* (pp. 1-5). IEEE.
- [42] Kiljander, J., D'elia, A., Morandi, F., Hyttinen, P., Takalo-Mattila, J., Ylisaukko-Oja, A., ... & Cinotti, T. S. (2014). Semantic interoperability architecture for pervasive computing and internet of things. *IEEE access*, 2, 856-873.
- [43] Jain, R. (2013). Networking protocols for internet of things. *St. Louis, MO: Washington University in St. Louis*.
- [44]. Bello, O., Zeadally, S., & Badra, M. (2017). Network layer inter-operation of Device-to-Device communication technologies in Internet of Things (IoT). *Ad Hoc Networks*, 57, 52-62.
- [45] Bassbouss, L., Telekom, A. D., Bauer, M. N., IoTecha, O., Alaya, B., Longstreth, M. S., ... & Gyr Davies, J. B. (2016). Semantic interoperability for the web of things. *ResearchGate Online resource*.
- [46] Jagatic, T. N., Johnson, N. A., Jakobsson, M., & Menczer, F. (2007). Social phishing. *Communications of the ACM*, 50(10), 94-100.
- [47] BERDAI, M. (2016). ÉTUDE COMPARATIVE DES PROTOCOLES ZIGBEE PRO ET ZIGBEE IP. [48] Lucio Silva, L. (2016). Internet of Things: Pros and cons of CoAP protocol solution for small devices.
- [49] Shelby, Z., Hartke, K., & Bormann, C. (2014). *The constrained application protocol (CoAP)* (No. rfc7252).

- [51] Ishaq, I., Hoebeke, J., Van den Abeele, F., Rossey, J., Moerman, I., & Demeester, P. (2014). Flexible unicast-based group communication for CoAP-enabled devices. *Sensors*, 14(6), 9833-9877.
- [52] Saad, L. B., Chauvenet, C., & Tourancheau, B. (2011, September). Simulation of the RPL Routing Protocol for IPv6 Sensor Networks: two cases studies. In *International Conference on Sensor Technologies and Applications SENSORCOMM 2011*. IARIA.
- [53] Moussa, M. (2014). *Vérification et configuration automatiques de pare-feux par Model Checking et synthèse de contrôleur* (Doctoral dissertation, École Polytechnique de Montréal)..
- [54] قريشي, & وسيلة. *Partage de données en environnements mobiles Ad hoc* (Doctoral dissertation, Université de Ouargla-Kasdi Merbah).
- [55] Tariq, M. A., Khan, M., Raza Khan, M. T., & Kim, D. (2020). Enhancements and challenges in coap—a survey. *Sensors*, 20(21), 6391.
- [59] Oryema, B., Kim, H. S., Li, W., & Park, J. T. (2017, January). Design and implementation of an interoperable messaging system for IoT healthcare services. In *2017 14th IEEE annual consumer communications & networking conference (CCNC)* (pp. 45-52). IEEE.
- [60] Ge, S. Y., Chun, S. M., Kim, H. S., & Park, J. T. (2016, January). Design and implementation of interoperable IoT healthcare system based on international standards. In *2016 13th IEEE annual consumer communications & networking conference (CCNC)* (pp. 119-124). IEEE.
- [61] Li, W., Jung, C., & Park, J. (2018). IoT healthcare communication system for IEEE 11073 PHD and IHE PCD-01 integration using CoAP. *KSII Transactions on Internet and Information Systems (TIIS)*, 12(4), 1396-1414.
- [62] Viel, F., Augusto Silva, L., Leithardt, V. R. Q., De Paz Santana, J. F., Celeste Ghizoni Teive, R., & Albenes Zeferino, C. (2020). An Efficient Interface for the Integration of IoT Devices with Smart Grids. *Sensors*, 20(10), 2849.
- [63] Tamboli, M. B., & Dambawade, D. (2016, May). Secure and efficient CoAP based authentication and access control for Internet of Things (IoT). In *2016 IEEE international conference on recent trends in electronics, information & communication technology (RTEICT)* (pp. 1245-1250). IEEE.
- [64] Krawiec, P., Sosnowski, M., Mongay Batalla, J., Mavromoustakis, C. X., & Mastorakis, G. (2018). DASCo: dynamic adaptive streaming over CoAP. *Multimedia Tools and Applications*, 77(4), 4641-4660.
- [65] ur Rahman, W., Choi, Y. S., & Chung, K. (2018, October). Quality Adaptation Algorithm for Streaming over CoAP. In *2018 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 1015-1018). IEEE.

- [66] Scott, T. L., & Eleyan, A. (2019, June). CoAP based IoT data transfer from a Raspberry Pi to Cloud. In *2019 International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1-6). IEEE.
- [67] Jan, S. R., Khan, F., Ullah, F., Azim, N., & Tahir, M. (2016). Using CoAP protocol for resource observation in IoT. *International Journal of Emerging Technology in Computer Science & Electronics*, ISSN: 0976, 1353.
- [68] Djamaa, B., Yachir, A., & Richardson, M. (2017). Hybrid CoAP-based resource discovery for the Internet of Things. *Journal of Ambient Intelligence and Humanized Computing*, 8(3), 357-372.
- [69] Djamaa, B., & Yachir, A. (2016). A proactive trickle-based mechanism for discovering CoRE resource directories. *Procedia Computer Science*, 83, 115-122.
- [70] Djamaa, B., Richardson, M., Aouf, N., & Walters, B. (2014). Towards efficient distributed service discovery in low-power and lossy networks. *Wireless Networks*, 20(8), 2437-2453.
- [71] Ugrenovic, D., & Gardasevic, G. (2015, November). CoAP protocol for Web-based monitoring in IoT healthcare applications. In *2015 23rd Telecommunications Forum Telfor (TELFOR)* (pp. 79-82). IEEE.
- [72] Fleury, G. (2007). Simulation à événements discrets.
- [73] Cenedese, A., Zanella, A., Vangelista, L., & Zorzi, M. (2014, June). Padova smart city: An urban internet of things experimentation. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014* (pp. 1-6). IEEE.

❖ Site Web:

- [5] <https://fr.wikipedia.org/wiki/VMware>.
- [6] https://www.researchgate.net/publication/335233372_STUDY_OF_INTERNET_OF_THINGS_AND_DEVELOPMENT_TOOLS_AND_TECHNOLOGY
- [26] https://www.javatpoint.com/http30/3/2022_18/13
- [27] https://www.thetechplatform.com/post/what-is-http-protocol-architecture-and-components-of-http_20/4/2022
- [28] <https://ducmanhphan.github.io/2019-02-05-HTTP-protocol/> 20/5/2022
- [30] <https://fr.wikipedia.org/wiki/PyCharm>.
- [41] <https://whatis5g.info/Energy-Consumption>.
- [56] https://www.tutorialspoint.com/internet_of_things/internet_of_things_tutorial.pdf.
- [57] <http://lacl.univ-paris12.fr/cegielski/serveur/ch3.pdf>
- [58] https://fr.wikipedia.org/wiki/CoAP#Applications_pratiques