HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INTERNATIONAL EDUCATION

SCIENCE AND TECHNOLOGY MAJOR

──────── \* ────────

**THESIS**

SUBMITTED FOR PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

**ENGINEER**

IN

**INGORMATION TECHNOLOGY**

**RESEARCH AND SIMULATION OF TSCH FOR WIRELESS SENSOR NETWORK**

Author: **Cung The Hung**

Class: LTU11B

Supervisor: **Msc.** **Banh Thi Quynh Mai**

HANOI, 01/2018

G**RADUATION THESIS DELEGATION SHEET**

**1. Student information:**

Full name: Cung The Hung

Mobile number: 0888514052 Email: cungthehung@gmail.com

Class: LTU11B

Hanoi University of Science and Technology

Time: from 01/ 09 /2017 to 08/01/2018

**2. Thesis target:**

Establishment of service quality levels for multi-access wireless sensor network time slot and frequency hopping. Build a wireless sensor network that collects data and sends data to the mobile gateway.

**3. Main tasks of graduation thesis:**

* Learn about Wireless Sensor Networks.
* Learn about multi-access by time slot frequency hopping.
* Find service levels for Wireless Sensor Networks.
* Building a Wireless Sensor Network for collecting data multiple access by time slot and frequency hopping using a mobile Gateway.
* Test performance and performance evaluation.

**4. Student commitment:**

I would like to commit that the graduation thesis is my own research under the guidance of Msc.Banh Thi Quynh Mai. All the results in the thesis are truthful and I don’t copy from other sources.

|  |  |
| --- | --- |
|  | *Hanoi, 06th Jan 2018*  Thesis author  Cung The Hung |

**5. Approval of Lecturer:**

|  |  |
| --- | --- |
|  | *Hanoi, 08th Jan 2018*  Supervisor  Msc. Banh Thi Quynh Mai |

# ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to all those who help me the academic supporting that I received during the time of my studying at the Hanoi University of Science and Technology.

I wish to acknowledge, first and foremost, my supervisor, Mrs. Banh Thi Quynh Mai - Lecturer of The Department of Communication and Computer Networks, School of Information and Communication Technology, Hanoi University of Science and Technology, for his guidance, caring and support throughout the completion of this thesis.

Finally, I owe a very important debt to my family and my friends for always believing in me and supporting my decisions.

**ABSTRACT OF THESIS**

Wireless data sensor networks are an essential requirement of major cities around the world. Within the framework of this project, I have introduced an architecture design for wireless data sensor networks. Particularly, building a wireless sensor network to collect data at the park, divided into small clusters, each cluster will have a relay node, which will have a mobile gateway to collect data in order to transmit to the processing center. Mobile gateways are selected as city buses so the speed of the bus depends on the traffic density in the road. The speed of the bus is devided into three types: fast, slow and medium. With these three types, we need three levels of service in the wireless sensor network to match the speed of the mobile gateway. Successfully built three levels of service quality in wireless sensor networks using multiple timeslot access and frequency hopping. Service quality levels introduce different latencies in the network, and consumption energy increases as latency decreases. However, the increase in power consumption is allowed for wireless sensor networks. Flexibly changing the quality of service levels does not always require the highest quality of service, but the quality of service will vary depending on the speed of the mobile gateway. network reduces power consumption. Experimental setup, performance measurement shows successful packet transfer rate is over 80%.

Table of Contents

[ACKNOWLEDGEMENTS 3](#_Toc503183057)

[LIST OF FIGURES 7](#_Toc503183058)

[LIST OF TABLES 8](#_Toc503183059)

[NOMENCLATURE 8](#_Toc503183060)

[CHAPTER I: INTRODUCTION 10](#_Toc503183061)

[**1.** **PROBLEM** 10](#_Toc503183062)

[**2.** **SOLUTION** 10](#_Toc503183063)

[CHAPTER II: THEORY AND TECHNOLOGY 11](#_Toc503183064)

[**1.** **WIRELESS SENSOR NETWORK (WSN)** 11](#_Toc503183065)

[**1.1.** **Characteristics of a WSN** 12](#_Toc503183066)

[**1.1.1.** **Power efficiency in wireless sensor networks** 12](#_Toc503183067)

[**1.1.2.** **Scalability in wireless sensor networks.** 13](#_Toc503183068)

[**1.1.3.** **Wireless sensor networks responsiveness:** 14](#_Toc503183069)

[**1.1.4.** **Reliability in wireless sensor networks** 15](#_Toc503183070)

[**1.1.5.** **Mobility in wireless sensor networks** 16](#_Toc503183071)

[**2.** **Standards for WSN** 17](#_Toc503183072)

[**2.1.** **IEEE 802.15.4** 17](#_Toc503183073)

[**2.1.1.** **Confined of IEEE 802.15.4** 18](#_Toc503183074)

[**2.1.2.** **IEEE 802.15.4e** 18](#_Toc503183075)

[**3.** **Multiple access mechanisms in wireless sensor networks** 20](#_Toc503183076)

[**3.1.** **Time Slotted Channel Hopping (TSCH)** 20](#_Toc503183077)

[**3.1.1.** **Slotframe structure and synchronization** 20](#_Toc503183078)

[**3.1.2.** **Channel hopping** 22](#_Toc503183079)

[**3.1.3.** **TSCH CSMA-CA algorithm** 23](#_Toc503183080)

[**3.2.** **ORCHESTRA** 25](#_Toc503183081)

[**3.2.1.** **Overall** 26](#_Toc503183082)

[**3.2.2.1.** **Orchestra Slots** 28](#_Toc503183083)

[**3.2.2.2.** **Orchestra Slotframes** 29](#_Toc503183084)

[**3.2.2.3.** **Scheduling Rules** 30](#_Toc503183085)

[**4.** **CONTIKI OPERATING SYSTEM AND SIMULATION COOJA.** 31](#_Toc503183086)

[**4.1.** **CONTIKI OPERATING SYSTEM.** 31](#_Toc503183087)

[**4.2.** **COOJA-SIMULATION TOOL** 32](#_Toc503183088)

[**5.** **CONLUSION** 33](#_Toc503183089)

# LIST OF FIGURES

|  |  |
| --- | --- |
| Figure | Description |
| *Figure 2.1* | *Wireless Sensor NetWork* |
| *Figure 2.2* | *Power efficiency in wireless sensor networks* |
| *Figure 2.3* | *Scalability in wireless sensor networks* |
| *Figure 2.4* | *Wireless sensor networks responsiveness* |
| *Figure 2.5* | *Reliability in wireless sensor networks* |
| *Figure 2.6* | *Mobility in wireless sensor networks* |
| *Figure 2.7* | *A TSCH slotframe on the 2.4GHz band. Each color represents a layer 2 (MAC) link between two devices*. |
| *Figure 2.8* | *slotframe (TSCH)* |
| *Figure 2.9* | *Timeslot (TSCH)* |
| *Figure 2.10* | *A sensor network with a tree-topology with a possible link schedule for data-collection.* |
| *Figure 2.11* | *Orchestra Schedule* |
| Figure 2.12 | *Illustration of the different Orchestra slot types, in a 4-node network,* |
| *Figure 2.13* | *Network protocol architecture* |
| *Figure 2.14* | *Cooja screenshot* |
| *Figure 3.1* | *Network Topology* |
|  |  |
|  |  |

# LIST OF TABLES

# NOMENCLATURE

|  |  |
| --- | --- |
| Acronym | Define |
| TSCH | Time Slotted Channel Hopping |
| IEEE | Institute of Electrical and Electronics Engineers |
| 6LoWPAN | IPv6 over Low power Wireless Personal Area Networks |
| RPL | Routing Protocol for Low-power and Lossy Networks |
| PAN | Personal Area Network |
| MAC | Medium Access Control |
| BE | Beacon Enabled |
| NBE | Non-Beacon Enabled |
| CSMA-CA | Carrier-sense multiple access with collision avoidance |
| SBS | Sender-based Shared Orchestra Slots |
| SBD | Sender-based Dedicated Orchestra Slots |
| ASN | Absolute Slot Number |
| TDMA | Time-division multiple access |
| QoS | Quality of Service |
| CS | Common Shared Orchestra Slots |
| RBS | Receiver-based Shared Orchestra Slots |

# CHAPTER I: INTRODUCTION

1. **PROBLEM**

Nowadays, Internet of Things and Wireless Sensor Network become the development tendency. The IEEE 802.15.4, which has made a considerable contribution to this development, set many standards for Physics and Mac suitable for the requirements of the Wireless Sensor Network. However, the IEEE 802.15.4 standard still lacks of latency, communication reliability, and energy efficiency. These limitations are not entirely consistent with the large-scale problem, requiring energy savings, real-time and high reliability requirements such as the smart city problem. To response to this problem, the IEEE 802.15.4e standard was introduced that overcomes the limitations of 802.15.4. One of the highlights of 802.15.4e is the definition of the Time Slotted Channel Hopping (TSCH) protocol for the MAC layer, with slot-based channel access and channel hopping mechanisms, the TSCH has several advantages that match the Smart City. The TSCH operates on a schedule, however, 802.15.4e does not specifically describe how to schedule it, so Orchestra is given as a scheduling method for TSCH. A model for smart cities is shown in the figure below (Figure 1.1). Specifically, in the city, there will be many network sensors to measure temperature, water quality, air quality ... located in lakes, parks or residential clusters. In each sensor network cluster, there will be a node acting as a coordinator, collecting data in a network cluster and as a control center. The city-running buses act as mobile gateways that collect data from the sensor network clusters throughout the city. When approaching a network clustering, the gateway will communicate with the cluster's coordinator to retrieve the data. At different times, different traffic density leads to the fast or slow bus speeds. As a result, the latency and packet transmission requirements in the sensor network clusters also change. Besides, depending on the time or different neccessaries, the frequency of data collection also needs to change. In order to meet these requirements, in a sensor network using the TSCH protocol at the MAC layer, we can influence the number of channels transmitted for the purpose of efficiently utilizing energy. This option is based on the linkage between the number of transmission channels and the performance parameters in the TSCH network, which will be discussed further below. As such, depending on the different needs, the sensor network clusters will respond at different service levels. In particular, at higher levels of service quality, network latency decreases, transmission reliability increases but more energy consumes and vice versa with lower service levels. For the reasons mentioned above, the problem posed in this project is: Change the number of channels in the TSCH and Orchestra wireless sensor networks to provide different levels of service quality.

1. **SOLUTION**

With the above problem, some issues need to be resolved in the following directions:

* Building sensor networks with TSCH and Orchestra: nodes in the network are installed overlapping protocol (protocol stack up to the UDP layer, using the TSCH protocol and Orchestra at the MAC layer. The nodes are divided into two types according to function: node coordinator and data collection node.
* Provide service levels consistent with the number of channels in the network.
* Select the way to control the service level for the entire network: The node coordinator receives external control information, and then creates and distributes the UDP control messages to all nodes in the network.

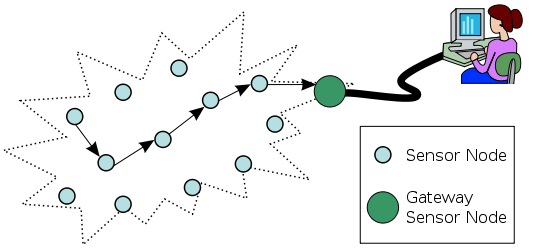
# CHAPTER II: THEORY AND TECHNOLOGY

1. **WIRELESS SENSOR NETWORK (WSN)**

WSN is spatially distributed [autonomous](https://en.wikipedia.org/wiki/Autonomous) [sensors](https://en.wikipedia.org/wiki/Sensor) to monitor physical or environmental conditions, such as [temperature](https://en.wikipedia.org/wiki/Temperature), [sound](https://en.wikipedia.org/wiki/Sound), [pressure](https://en.wikipedia.org/wiki/Pressure), etc. and to cooperatively pass their data through the network to a main location. The more modern networks are bi-directional, also enabling control of sensor activity. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance; today such networks are used in many industrial and consumer applications, such as industrial process monitoring and control, machine health monitoring, and so on.

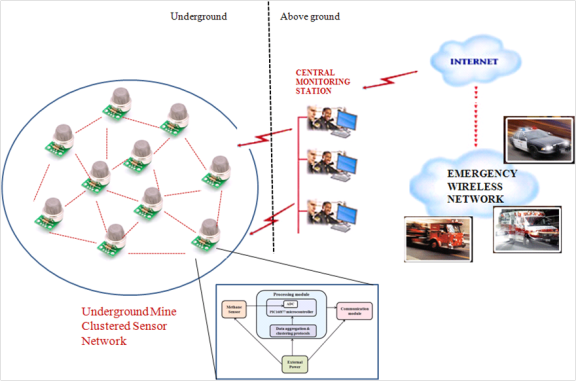
The WSN is built of "nodes" – from a few to several hundreds or even thousands, where each node is connected to one (or sometimes several) sensors. Each such sensor network node has typically several parts: a [radio](https://en.wikipedia.org/wiki/Radio) [transceiver](https://en.wikipedia.org/wiki/Transceiver) with an internal [antenna](https://en.wikipedia.org/wiki/Antenna_(radio)) or connection to an external antenna, a [microcontroller](https://en.wikipedia.org/wiki/Microcontroller), an electronic circuit for interfacing with the sensors and an energy source, usually a [battery](https://en.wikipedia.org/wiki/Battery_(electricity)) or an [embedded](https://en.wikipedia.org/wiki/Embedded_system) form of [energy harvesting](https://en.wikipedia.org/wiki/Energy_harvesting). A [sensor node](https://en.wikipedia.org/wiki/Sensor_node) might vary in size from that of a shoebox down to the size of a grain of dust, although functioning "motes" of genuine microscopic dimensions have yet to be created. The cost of sensor nodes is similarly variable, ranging from a few to hundreds of dollars, depending on the complexity of the individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and communications bandwidth. The topology of the WSNs can vary from a simple [star network](https://en.wikipedia.org/wiki/Star_network) to an advanced [multi-hop](https://en.wikipedia.org/wiki/Mesh_networking) [wireless mesh network](https://en.wikipedia.org/wiki/Wireless_mesh_network). The propagation technique between the hops of the network can be [routing](https://en.wikipedia.org/wiki/Routing) or [flooding](https://en.wikipedia.org/wiki/Flooding_algorithm).

The sensor nodes are usually scattered in the environment and connected to the root node (sink/gateway).



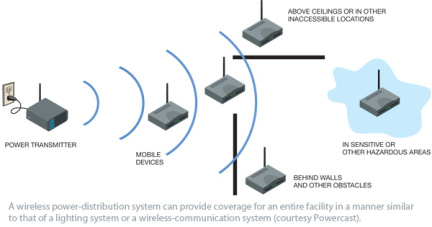
*Figure 2.1: Wireless Sensor NetWork*

* 1. **Characteristics of a WSN**
     1. **Power efficiency in wireless sensor networks**

****

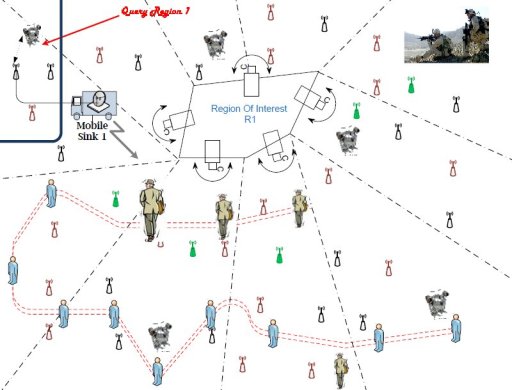
*Figure 2.2: Power efficiency in wireless sensor networks*

* Power efficiency as the name sugMobility is the networks ability to handle mobile nodes and changeable data paths.
* The way the design goes it is necessary for wireless sensor network to be highly responsive in order for it to deal Mobility is the networks ability to handle mobile nodes and changeable data paths.
* The way the design goes it is necessary for wireless sensor network to be highly responsive in order for it to deal with mobility. As a result of which it becomes harder to design a large scale as well as mobile wireless sensor network.with mobility. As a result of which it becomes harder to design a large scale as well as mobile wireless sensor network.gests is the efficiency levels of the device in question. Here it is the ability of the device to consume less power to do more by operating under extremely low power levels.
* This is a very important factor; specially so since it often happens with wireless sensor networks that the sensor nodes are remotely located with no proper access to a power point.
* Thus these devices are usually built with the ability to work from a power source other than direct electricity. The optimal method of design would be by reducing the duty cycle of each node.
* Additionally wireless sensors are put to sleep as well to save power and thus are un-responsive to neighbor communication.
  + 1. **Scalability in wireless sensor networks.**

****

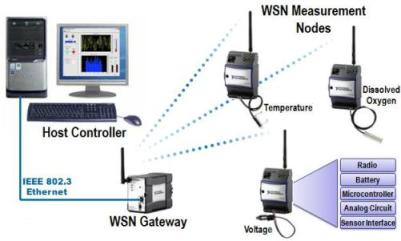
*Figure 2.3: Scalability in wireless sensor networks*

* The ability for a network to grow in terms of the number of nodes attached to the wireless sensor network with causing excessive overhead can be termed as its scalability.
* The basic implementation of such a network usually consists of only a handful of nodes and it is important that they provide support for more as well.
* There are many kinds of networks; like in an ad hoc network the network is formed without any predetermined topology and thus nodes wishing to communicate with other nodes need to generate more packets than its data packets. Thus as the size of the network grows more packets will be required.
* There is a higher chance of communication links being broken as the network size increases. Here only small amount of bandwidth will be left for the application data transmission as the network grows.
  + 1. **Wireless sensor networks responsiveness:**

****

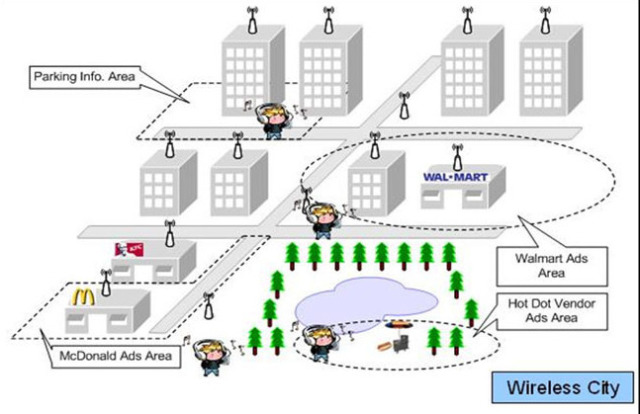
*Figure 2.4: Wireless sensor networks responsiveness*

* The ability of the network to quickly adapt itself to change in the topology is considered its responsiveness.
* There are however downfalls to have a highly responsive network; compromises need to be made. The latency of packet delivery in dynamic environment as well as scalability will decrease in highly responsive network.
  + 1. **Reliability in wireless sensor networks**

****

*Figure 2.5: Reliability in wireless sensor networks*

* Any network needs to be reliable; that would be the basic requirement. You do require reliable data transmission in a state of continuous change of the network structure.
* There is usually a inverse relationship between scalability and reliability in ad hoc wireless networks. This is because as the number of nodes in the network increases it becomes harder to maintain the reliability.
* If the network is highly scalable and is scaled in to a bigger network than what it was originally designed to be then this will place a strain on the reliability of data transmission and the breaking point will show up sooner.
  + 1. **Mobility in wireless sensor networks**

****

*Figure 2.6: Mobility in wireless sensor networks*

* Mobility is the networks ability to handle mobile nodes and changeable data paths.
* The way the design goes it is necessary for wireless sensor network to be highly responsive in order for it to deal with mobility. As a result of which it becomes harder to design a large scale as well as mobile wireless sensor network.

1. **Standards for WSN**
   1. **IEEE 802.15.4**

IEEE 802.15.4 is a technical standard which defines the operation of low-rate wireless personal area networks (LR-WPANs). It specifies the physical layer and media access control for LR-WPANs, and is maintained by the IEEE 802.15 working group, which defined the standard in 2003. It is the basis for the ZigBee,[2] ISA100.11a,[3] WirelessHART, MiWi, SNAP, and Thread specifications, each of which further extends the standard by developing the upper layers which are not defined in IEEE 802.15.4. Alternatively, it can be used with 6LoWPAN, the technology used to deliver the IPv6 version of the Internet Protocol (IP) over WPANs, to define the upper layers the [IPv6](https://en.wikipedia.org/wiki/IPv6) version of the [Internet Protocol](https://en.wikipedia.org/wiki/Internet_Protocol) (IP) over WPANs, to define the upper layers.

* + 1. **Confined of IEEE 802.15.4**

IEEE 802.15.4 defines the physical layer and the MAC (Medium Access Control) layer in the stratified network architecture and is considered the reference standard for wireless sensor networks (WSNs). In fact, products that comply with this standard are still very popular. Many studies examine and evaluate the performance of IEEE 802.15.4 in WSANs (Wireless Sensor and Actuator Networks), but also point out some of its limitations:

* No latency limitation: channel access method is based on CSMA-CA algorithm.
* Limitations on communication reliability.
* No anti-jamming mechanism.
* Consume more energy

Due to the above limitations, 802.15.4 is not suitable for some largescale application scenarios, requiring efficient, real-time energy efficiency and high reliability. Examples of industrial applications, smart cities, ...

* + 1. **IEEE 802.15.4e**

The IEEE 802.15 Task Group 4e was chartered to define a MAC amendment to the existing standard 802.15.4-2006. The intent of this amendment was to enhance and add functionality to the 802.15.4-2006 MAC to a) better support the industrial markets and b) permit compatibility with modifications being proposed within the Chinese WPAN.

On February 6, 2012 the IEEE Standards Association Board approved the IEEE 802.15.4e MAC Enhancement Standard document for publication.

The approximate publication date of the IEEE Std. 802.15.4e-2012 standard is on or about March 28, 2012.

**IEEE 802.15.4e introduces the following general functional enhancements:**

* *Low Energy (LE).* This mechanism is intended for applications that can trade latency for energy efficiency. It allows a node tope rate with a very low duty cycle (e.g., 1% or below), while appearing to be always on to the upper layers. This mechanism is important for enabling the Internet of Things paradigm as Internet protocols have been designed assuming that hosts are always on.
* *Information Elements (IE).* It is an extensible mechanism to exchange information at the MAC sublayer.
* *Enhanced Beacons (EB).* Enhanced Beacons are an extension of the 802.15.4 beacon frames and provide a greater flexibility. They allow to create application specific frames, by including relevant IEs.
* *Multipurpose Frame.* This mechanism provides a flexible frame format that can address a number of MAC operations. It is based on IEs.
* *MAC Performance Metric*. It is a mechanism to provide appropriate feedback on the channel quality to the networking and upper layers, so that appropriate decision can be taken. For instance, the IP protocol may implement dynamic fragmentation of datagrams depending on the channel conditions.
* *Fast Association (FastA).* The 802.15.4 association procedure introduces a significant delay in order to save energy. For timecritical application latency has priority over energy efficiency. Therefore, the FastA mechanism allows a node to associate in a reduced amount of time.

**MAC behavior modes:**

* Time Slotted Channel Hopping (TSCH). It targets application domains such as industrial automation and process control, providing support for multi-hop and multi-channel communications, through a TDMA approach.
* Deterministic and Synchronous Multi-channel Extension (DSME). It is aimed to support both industrial and commercial applications with stringent requirements in terms of timeliness and reliability. To this end, it combines contention-based and timedivision medium access, and offers two different channel diversity modes. It is specifically designed for multi-hop and mesh networks.
* Low Latency Deterministic Network (LLDN). Designed for singlehop and single-channel networks, it is intended for factory automation, where applications require very low latency.
* Asynchronous multi-channel adaptation (AMCA). It is targeted to application domains where large deployments are required, such as smart utility networks, infrastructure monitoring networks, and process control networks.
* Radio Frequency Identification Blink (BLINK). It is intended for application domains such as item/people identification, location and tracking. Specifically, it allows a node to communicate its ID to other nodes without prior association and without acknowledgement. BLINK packets are generally sent by ‘transmit only’ devices through the Aloha protocol.

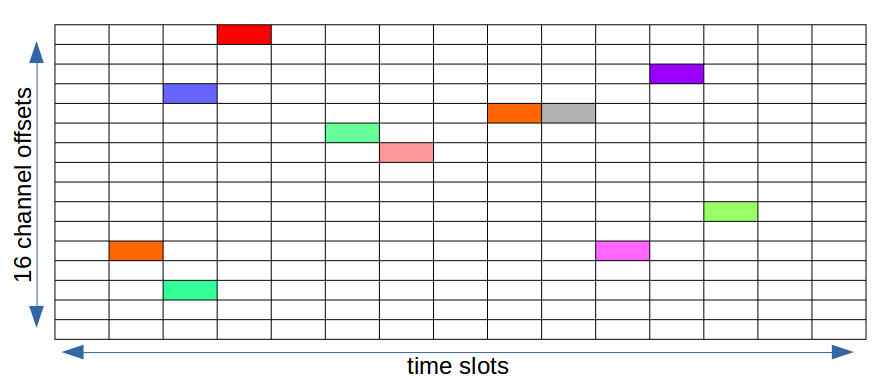
1. **Multiple access mechanisms in wireless sensor networks**
   1. **RPL**

RPL is an oriented distance-vector routing protocol that organizes nodes in a Destination Oriented Directed Acyclic Graph (DODAG) structure. Directed acyclic graph (DAG): is a finite [directed graph](https://en.wikipedia.org/wiki/Directed_graph) with no [directed cycles](https://en.wikipedia.org/wiki/Cycle_graph#Directed_cycle_graph). That is, it consists of finitely many [vertices](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) and [edges](https://en.wikipedia.org/wiki/Edge_(graph_theory)), with each edge directed from one vertex to another. There is not the path that starts at any vertex *v* and ends at *v*. Destination Oriented Directed Acyclic Graph (DODAG) is a DAG rooted at a single destination. RPL network is flexible; in other word, it has three flowing capabilities. The first capability is self – configuration. When each end node is powered on, it listens for its neighbor nodes. If it finds one or more, it issues a request to join the network and provided that it meets admission criteria. After the end node joins the network, paths (or routes) will be automatically formed by the end node as the information that it transmits gets relayed by neighboring nodes until it reaches the central node. The next capability is self – healing. Loss of one or more nodes does not necessarily affect operation of the network. If a node in a network fails, messages are sent via other nodes. The final capability is scaling. RPL network can be simply expanded by adding more end nodes.

* Node rank: Each node is attached a rank. Node rank deﬁnes a node’s relative position within a DODAG with respect to the DODAG root. Rank strictly increases in the down direction and strictly decreases in the up direction. The exact way rank is computed depending on the Objective Function (OF).
* RPL Instance: Deﬁnes Optimization Objective when forming paths towards roots based on one or more metrics. Metrics may include both Link properties (Reliability, Latency) and Node properties (Powered or not).
* RPL Control messages:
  + DAG Information Object (DIO): carries information that allows a node to discover an RPL Instance, learn its conﬁguration parameters and select DODAG parents and maintain the DODAG.
  + DAG Information Solicitation (DIS): probe its neighborhood.
  + Destination Advertisement Object (DAO): propagate destination information upward.
* DODAG construction: Initially, the root advertises the information about the graph using the DIO message. The nodes in the listening vicinity (neighboring nodes) of the root will receive, process DIO message and makes a decision based on certain rules (according to the objective function, DAG characteristics, advertised path cost and potentially local policy) whether to select the DIO sender as its parent. Once the node has joined the graph, it has a route toward its parent. The node computes its rank and replies with a DAO message with has prefix information to its parent to inform its participation. If configured to act as a router, it broadcasts a DIO message containing the new graph information to its neighboring peers. If a neighbor is a “leaf node”, it simply joins the graph and does not send any DIO message. If a neighbor selects the node as its parent, it also computes its rank and replies a DAO message. The node integrates prefix information of received DAO messages and send a DAO message to its parent node. All of the neighboring peers will repeat this process until all of the nodes join the DODAG. The node that has not received any DIO messages and has not joined any DODAGs can request DODAG information by sending DIS message periodically to its neighbors. In this way, each node of the graph has a routing entry towards its parent and the leaf nodes can send a data packet all the way to root of the graph by just forwarding the packet to its immediate parent.
* Downward routing with storing mode: all RPL nodes maintain next hop addresses for the nodes in their sub – DODAG. Each node send unicast DAO messages to its parents to advertise routes reaching target node. Upon receiving a DAO message, a node locally stores in its routing table the address of the sender as the next hop node to reach the advertised targets and also generates a new DAO message and transmits it to its parents to ensure that routing information propagates upward in the network.
* Object function: allows RPL to optimize or satisfy the routing metric of a path. For example, if an RPL instance uses an objective function that minimizes hop-count, RPL will select paths with minimum hop count.
  + ETX object function (ETXOF): The ETX metric describes the expected number of transmissions required to successfully transmit and acknowledge a packet on a wireless link. ETXOF allows RPL to find a minimum – ETX path from the nodes to a root in the DAG instance. The minimum – ETX path is the path that requires the least number of packet transmissions per packet delivery to the DAG root. Thus, minimum – ETX paths are generally also the most energy- efficient paths in the network. Firstly, ETXOF compute ETX path metric based on ETX link metric. The min\_path\_etx is the ETX path metric of the path from a node through its preferred parent to the root computed at the last parent selection. Root sets the variable min\_path\_etx to MIN\_ETX\_PATH\_CONST (= 1). A non – root nodes computes the ETX metric for the link to a candidate neighbor and the min\_path\_etx computed at that neighbor. That neighbor send a DIO message containing min\_path\_etx to its neighbors. If the ETX metric of the link to a neighbor is not available, the ETX Path metric for the path through that neighbor SHOULD be set to INFINITY. The ETX Path metric corresponding to a neighbor must be re – computed each time the ETX metric of the link to the candidate neighbor is updated or a node receives a new min\_path\_etx advertisement from the candidate neighbor. In the subsequent stage, the node select its preferred parent. A candidate neighbor is selected as the node’s parent if the ETX path metric corresponding to that neighbor is smaller than the ETX path metric corresponding to the rest of the neighbors, except following cases:
    - If the ETX Path metric of the current preferred parent is greater than the smallest ETX Path metric by less than PARENT\_SWITCH\_ETX\_THRESHOLD (= 0.5), the node MAY continue to use the current preferred parent.
    - The node is a floating root and it does not have parent.
    - If the ETX metric for a link to a neighbor is greater than MAX\_ETX\_LINK\_CONST (= 10.0), the neighbor is not selected.
    - If min\_path\_etx computed at a node is greater than MAX\_ETX\_PATH\_CONST (= 200.0), the node may declare itself as a floating root. If the configuration disallows a node to be a floating root and no neighbors are discovered, the node does not have a preferred parent, and must set min\_path\_etx to INFINITY.
    - If the ETX path metric corresponding to multiple candidate neighbors are equal, a node may use a different objective function to select the preferred parent. If the current preferred parent is one of these candidate neighbors, the parent is continuously used
* Routing from any node to another: is done by ﬁrst routing up to a common ancestor (along decreasing ranks), then down to the destination (following the routing tables).
  1. **Time Slotted Channel Hopping (TSCH)**

TSCH is used by Low-Power devices to communicate using a wireless link. It is designed for low-power and lossy networks (LLNs) and aims at providing a reliable Media access control layer.

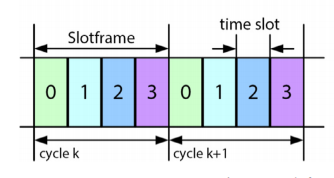
TSCH can be seen as a combination of Time division multiple access and Frequency-division multiple access mechanisms as it uses diversity in time and frequency to provide reliability to the upper network layers.



*Figure 2.7: A TSCH slotframe on the 2.4GHz band. Each color represents a layer 2 (MAC) link between two devices*.

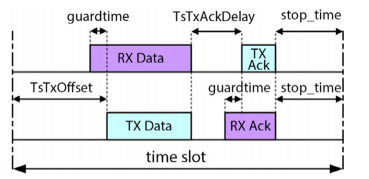
* + 1. **Slotframe structure and synchronization**

In the TSCH mode nodes synchronize on a periodic slotframe consisting of a number of timeslots (Fig 2.8).



*Figure 2.8: slotframe (TSCH)*

Each timeslot allows a node to send a maximum-size data frame and receive the related acknowledgement (Fig 2.9).



*Figure 2.9: Timeslot (TSCH)*

Inside a timeslot, data packets are transmitted exactly after TsTxOffset μs from the beginning of the timeslot itself. However, to allow for slight desynchronization, the receiver node starts listening the channel GuardTime μs before. In addition, if the reception of the packet does not begin within GuardTime μs after TsTxOffset, the node turns off its radio to save energy. This mechanism requires nodes to never be desynchronized for more than GuardTime μs, so as to be able to communicate. Anyway, due to differences in manufacturing, temperature and supply voltage, clocks of different nodes typically pulse at a slightly different frequency, resulting in “clock drift”. Hence, nodes need to periodically re-synchronize. To this end, each node is associated to a time-source neighbor, to which it must remain synchronized over time (although the 802.15.4e standard does not detail how such a neighbor must be selected).

There are two ways for a node to re-synchronize:

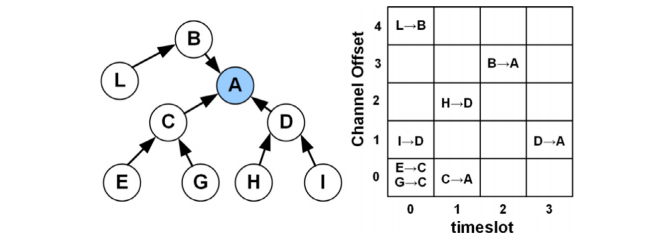
* Frame-based synchronization and ACK-based synchronization. In Frame-based synchronization, every time a node receives a data packet from its time source neighbor, it takes note of the instant the reception started. Then, since TsTxOffset is known, it shifts its slot boundaries to match those of its time source.
* Similarly, in ACKbased synchronization, every time a node sends a packet to its time source neighbor, the latter takes note of the instant it started receiving the packet, and inserts the obtained timestamp in a field of the acknowledgment. Once again, the sending node uses this value to realign its clock.
  + 1. **Channel hopping**

One of the main characteristics of TSCH is multi-channel communication, based on channel hopping. Initially, 16 different channels (frequency range from 2.4-2.4935 GHz) are available for communication. However, some of these frequencies could be blacklisted (because of low quality communication) and, hence, the total number of channels Nchannels available for channel hopping may be lower than 16. In TSCH, a link between communicating nodes can be represented by a pair specifying the timeslot in the slotframe and the channel offset used by the nodes in that timeslot. Let [n, channelOffset] denote a link between two nodes. Then, the frequency f to be used for communication in timeslot n of the slotframe is derived as follows.

*f = F[(ASN + channelOffset) % N\_channels]*

where ASN is the Absolute Slot Number, defined as the total number of timeslots elapsed since the start of the network. The ASN increments globally in the network, at every timeslot, and is thus used by nodes as timeslot counter. Function F can be implemented as a lookup table.

Thanks to the multi-channel mechanism, several simultaneous communications can take place in the same timeslot, provided that they use different channel offsets. Also, Eq. (1) implements the channel hopping mechanism by returning a different frequency for the same link at different timeslots. This assures that over time all the available channels are used for communications in a link and, hence, allows to mitigate the negative effect of external interference.



*Figure 2.10: A sensor network with a tree-topology with a possible link schedule for data-collection.*

Fig. 2.10 shows a possible link schedule for data collection in a simple network with a tree topology. We have assumed that the slotframe consists of 4 timeslots and there are only 5 channel offsets available. We can see that, thanks to the multi-channel approach used by TSCH, 8 transmissions have been accommodated in a time interval corresponding to 4 timeslots. In the allocation shown in Fig. 2.10 all links but one are dedicated links, i.e., allocated to a single node for transmission. TSCH also allows shared links, i.e., links intentionally allocated to more than one node for transmission. This is the case of the link [0,0] allocated to both nodes E and G.

* + 1. **TSCH CSMA-CA algorithm**

Since shared links can be accessed simultaneously by more than one transmitter, collisions may occur that result in a transmission failure. To reduce the probability of repeated collisions, the standard defines a CSMA-CA retransmission algorithm.

Upon receiving a data frame destined to node r, a sender node s waits for the arrival of the first (dedicated or shared) link assigned to (s, r), and, then, transmits its data frame. If a shared link was used and the transmission was unsuccessful (i.e. the acknowledgment was not received), very likely a collision occurred. Hence, the CSMA-CA algorithm is executed by node s to avoid repeated collisions. Specifically, the following steps are performed by node s.

* A set of state variables is initialized, namely the number of retransmissions carried out for the on-going frame (NB = 0) and the backoff exponent (BE = macMinBE).
* A random number w ∈ [0, 2BE − 1] is generated.
* . The frame retransmission is deferred for w shared links with destination r, or until a dedicated link with destination r is encountered.
* If the retransmission occurs in a shared link and it is successful (i.e. the acknowledgement is received), the backoff exponent BE is reset to macMinBE and the algorithm terminates. Instead, if the transmission is unsuccessful, state variables are updated as follows: NB = NB + 1, BE = min(BE + 1, macMaxBE). Finally, if the number of retransmissions for the current frame has exceeded the maximum allowed value (i.e. NB > macMaxFrameRetries) the frame is dropped; otherwise the algorithm falls back to step 2.

If the frame retransmission is carried out in a dedicated link, and it is successful, BE is reset to macMinBE, unless there are other frames, destined to the same receiver, ready for transmission. In the latter case the value of BE is left unchanged. Here we emphasize the differences between the original 802.15.4 CSMA-CA algorithm and the new TSCH CSMA-CA algorithm.

* **Backoff mechanism**. In the original 802.15.4 CSMA-CA each node with a packet ready for transmission waits for a random backoff time before trying to transmit it. The goal is to avoid collisions among nodes starting the execution of the CSMA-CA algorithm at the same time. Conversely, in TSCH CSMA-CA the backoff mechanism is activated only after the node has experienced a collision, i.e., it is used to avoid repeated collisions.
* **Backoff unit duration**. Both the 802.15.4 CSMA-CA and the TSCH CSMA-CA define a backoff unit. In both algorithms a node waits for a random number of backoff units before trying to retransmit a packet. However, while in the original 802.15.4 CSMA-CA the backoff unit is equal to 320μs, in TSCH the backoff unit corresponds to a shared slot. Using a slot as backoff unit assures that a node can experience a collision in a shared slot only if other nodes access the same slot. This is not true in the original 802.15.4 CSMA-CA where, in general, a packet can collide also with packets transmitted at a later time.
* **Clear Channel Assessment (CCA).** In the 802.15.4 CSMA-CA each node performs a CCA, to check the channel state, before performing a packet transmission. This is to avoid a collision with an ongoing transmission. In TSCH, CCAs are not used to prevent collisions among nodes, since all nodes are synchronized and no transmissions can be ongoing when a CCAs is performed. Conversely, the goal is to avoid transmitting a packet if a strong external interference is detected. In addition, in TSCH CSMA-CA, CCAs are optional.
* **Packet dropping**. In the original 802.15.4 CSMA-CA a packet is dropped by the sender if it has found the channel busy for macMaxCSMABackoffs consecutive times. This parameter is not used by TSCH. In TSCH CSMA-CA a packet is dropped only if it reaches the maximum number of retransmissions (specified by the macMaxFrameRetries parameter).
  1. **ORCHESTRA**

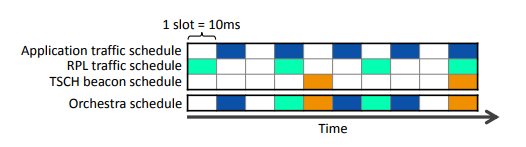
Orchestra is a new approach to scheduling in TSCH+RPL networks. Orchestra is radically different from existing scheduling solutions in that it does not involve any extra central entity, negotiation, signaling, nor multi-hop path reservation among nodes. Instead, nodes maintain their own schedule locally and autonomously, based on their RPL neighbors and parents. As a result, Orchestra makes TSCH as flexible as asynchronous MAC layers, and able to support random-access traffic.

An Orchestra schedule contains different slotframes of different lengths. Each slotframe is dedicated to a particular type of traffic: TSCH beacons, RPL signaling traffic or application data. Nodes select slots using scheduling rules which reduces contention drastically, or in certain cases eliminates contention. This makes Orchestra particularly appealing for low-power IPv6 scenarios where different applications generate event-based data, without any predefined (e.g., periodic) traffic pattern.

A concrete example Orchestra schedule contains:

* A dedicated broadcast slot from every node to its children for TSCH beacons, repeating every X slots;
* A slot common for all nodes in the network for broadcast+unicast for RPL signaling (DIO, DIS, DAO), repeating every Y slots;
* A dedicated unicast slot from every node to its RPL preferred parent, repeating every Z’ slots;
* N dedicated unicast slots from every node to each of its children, repeating every Z” slots.

Orchestra uses slotframe lengths which are mutually prime, ensuring the slots overlap each other evenly, without unintended synchronization effects. The key is that we select the time and channel offset of the every slot as a function of the sender’s or the receiver’s identifier (MAC address or a unique network node ID). Depending on the scheduling rules, Orchestra can either attain very low levels of contention, or operate contention-free.

****

*Figure 2.11: Orchestra Schedule*

* + 1. **Overall**

In Orchestra, nodes adapt their schedule by exploiting information from the RPL topology, and following a set of scheduling rules. This results in periodic activity patterns, with slotframes and slots assigned to different traffic planes such as TSCH beacons, RPL signaling, or application data.

* **Network Bootstrap**. When switched on, a node joins the TSCH network by listening until it receives a Enchanced Beacon (EB), either from the PAN coordinator or another node. After synchronizing to that EB, the node runs Orchestra. A viable Orchestra setup requires slots for sending and receiving packets to/from any neighbor. This emulates an always-on link to all neighbors, allowing RPL nodes to discover their neighbors and build a topology
* **TSCH-RPL Topology Mapping.** Orchestra consistently uses the node’s RPL preferred parent as its TSCH time source neighbor2. As the RPL topology evolves and parent switches occur, nodes update their TSCH time source accordingly. This yields a loop-free timing structure (a tree in this case), taking advantage of RPL’s built-in loop avoidance mechanism. Furthermore, we use the RPL rank as TSCH join priority, as defined in the 6TiSCH architecture [33]. Doing so, we also take advantage of the RPL mechanisms for gradient convergence and stability. In case a node loses synchronization, it also leaves the RPL network, ensuring a clean slate bootstrap after re-joining.
* **TSCH Time Synchronization.** TSCH time synchronization happens on any packet (or ACK) from the time source neighbor. In Orchestra, time synchronization happens primarily through periodic TSCH and RPL beacon transmission. This is efficient as a single broadcast message allows all children to update their clocks. Whenever a node has not communicated with its time source neighbor for a given duration (we use 12 s), it sends a unicast keepalive message. The packet is re-sent until acknowledged, and re-synchronization is done by using the timing information embedded in the IEEE802.15.4e enhanced ACK.
* **Routing-aware Scheduling.** Throughout the network lifetime, Orchestra installs and updates TSCH schedules by using information from the routing layer. RPL runs unmodified, with slots being set up automatically as the topology evolves, ensuring network connectivity and allowing upper layers to run transparently. A basic example is where Orchestra maintains a dedicated slot for parent to child communication, repeating at a fixed period (e.g., 1 s), at a time offset and channel offset selected from the parent’s unique node ID. Whenever a child switches parent, it updates its slot to match the new parent’s node ID.
  + 1. **Scheduling**

Orchestra runs deployment-specific scheduling rules that describes how to maintain TSCH slotframes and slots as a function of the routing topology.

* + - 1. **Orchestra Slots**

We identify four main types of slots in Orchestra: common shared slots, receiver-based shared slots, sender-based shared slots, and sender-based dedicated slots. Orchestra are dynamically mapped at runtime into 0, 1, or multiple TSCH slots. The different types of Orchestra slots are illustrated in Figure 2.12, in a 4-node network (Figure 2.12) and showing only the slots for child-to-parent traffic.

* Common Shared Orchestra Slots (CS). CS Orchestra slots consist in one shared slot used by all nodes in the network for both Rx (reception) and Tx (transmission), as illustrated in Figure 2.12b. The slot is installed at fixed coordinates (time and channel offset), resulting in a behavior similar to slotted ALOHA. This emulates an always-on link, allowing RPL to discover neighbors and run seamlessly. Note that TSCH uses an exponential back-off to resolve contention in shared slots, triggered whenever a unicast transmission is unacknowledged.
* Receiver-based Shared Orchestra Slots (RBS). RBS are assigned for communication between two neighbors, at coordinates (time and channel offset) derived from properties of the receiver. At every node, a RBS Orchestra slot results in one Rx slot (coordinates based on the node), and one Tx slot per neighbor (coordinates based on the neighbor). To calculate slot coordinates, one can use a hash of the node’s MAC address, modulo the slotframe length, or exploit unique node identifiers when available.
* A typical example is for child-to-parent communication: nodes listen for any traffic in one slot, and children maintain a transmit slot towards their parent. As nodes switch parent, they update their transmit slot autonomously. Because several nodes may install slots towards the same receiver, contention may arise in such slots. For instance in Figure 2.12c, #3 and #4 contend to send to their parent #2, using standard TSCH back-off
* Sender-based Shared Orchestra Slots (SBS). SBS are similar to RBS, except that the slot coordinates are obtained from properties of the sender node rather than the receiver. At every node, a SBS Orchestra slot results in one Rx slot per neighbor (coordinates based on the neighbor) and a single Tx slot (coordinates based on sender node). This results in higher energy consumption than RBS (Tx slots cost nothing when there is no traffic, whereas Rx slots always require a wakeup), but can also help decrease contention by avoiding per-receiver slot assignement.

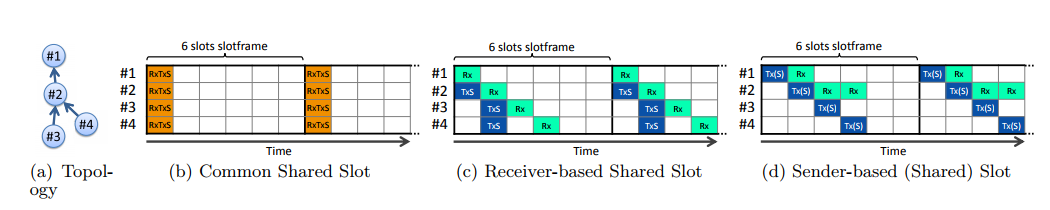
****

Figure 2.12: *Illustration of the different Orchestra slot types, in a 4-node network, showing child-to-parent slots. Slot properties are also shown: Reception (Rx), Transition (Tx), Shared (S). In common shared slots, all nodes wakeup up simultaneously to receive or transmit in a contention-based manner. In receiver-based slots, nodes have their own receive slot (here based on node ID), and their children contend when sending to them. In sender-based slots, nodes have their own transmit slot, and their parent wakes up to receive from them.*

* + - 1. **Orchestra Slotframes**

Orchestra manages several slotframes at every node, each of which is assigned to a particular traffic plane, e.g., TSCH beaconing, routing traffic, application. Slotframes consist of a set of slots, with properties defined by simple scheduling rules. The slotframes repeat at periods that are mutually prime, ensuring they cycle independently. In case slots from different slotframes overlap, the slot in the highest priority slotframe takes precedence3.

The length of a slotframe introduces a trade-off in traffic capacity, network latency and energy consumption.

* **Traffic Capacity**. Shorter slotframes have their slots repeat more often, resulting in higher traffic capacity. Orchestra’s approach is to over-provision TSCH in order to support non-deterministic traffic, and the slotframe length is the primary way to control the amount of over-provisioning for a given traffic plane.
* **Network Latency.** The per-hop latency on a given traffic plane is basically proportional to the length of the slotframe for this particular traffic plane.
* Energy Consumption. Similarly, the shorter the slotframe, the more often nodes have to wake up to listen or transmit, resulting in higher energy baseline.
  + - 1. **Scheduling Rules**

Orchestra maintains its schedules using simple scheduling rules, described in this section. Scheduling rules are a set of TSCH slotframes and slots enhanced with a number of Orchestra-specific properties. Some of the slotframe and slot properties are per IEEE802.15.4e (labeled std), other include extensions to standard properties (ext), or are introduced by Orchestra (new).

The properties of an Orchestra slotframe S are:

* **Handle (std).** A unique positive integer for both identifi- cation and priority. The smaller it is, the higher the priority.
* **Length (ext).** The number of slots in the slotframe. Must be mutually prime with all other slotframe lengths in the network.
* **Traffic Filter (new)**. The trafficplane the slotframe is intended for. Filters packet properties (e.g., unicast, broadcast) and protocols (e.g., TSCH, RPL).

Slotframes are made of Orchestra slots, each mapped into 0, 1 or multiple TSCH slots depending on the current TSCH and RPL state. An Orchestra slot can for instance be reserved for communication with all TSCH time sources, RPL children, or the current RPL preferred parent. The properties of a slot are:

* **Neighbors (new).** The neighbor or set of neighbors the Orchestra slot is to be instantiated for, such as the RPL preferred parent or all RPL children. The resulting TSCH slots are updated automatically whenever changes occur in the TSCH or RPL state.
* **Coordinates (ext).** The time and channel offset within the slotframe. Can either be fixed or a variable such as a node ID a hash of the neighbor MAC address.
* **Options (std).** Standard TSCH options. Includes: Rx (reception), Tx (transmission), S (shared), defining what the slot can be used for, and if it is shared or dedicated.

1. **CONTIKI OPERATING SYSTEM AND SIMULATION COOJA.**
   1. **CONTIKI OPERATING SYSTEM.**

Contiki is an open operating system designed specifically for applications with limited memory such as sensor nodes in wireless networks. Contiki is built on the C language, the open operating system with a small capacity and is based on the control of events. Contiki offers multi-tasks in order of priority. Contiki configuration occupies 2kB 40kB RAM and ROM memory .. Contiki is initiated research since 2001, released the first version in 2003, the business card version according to many changes, additions and outgrown Career has been applied in many practical projects...

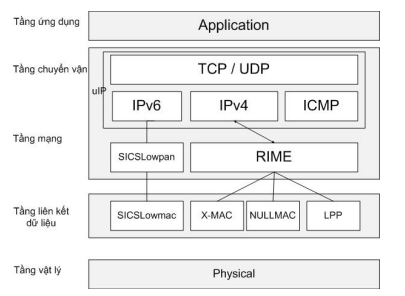
**Characteristics of Contiki operating system:**

* Contiki is divided into multiple modules to operate independently. So applications can use the module dynamically and load only the necessary modules.
* Active control events mechanism to reduce energy and limited consumption of storage space needed.
* Able to use the IP in sensor networks through uIP stack is built on TCP / IP.
* There are modules allows estimating and managing energy efficiently.
* Easy interaction protocols between layers and the nodes in the network
* Using Rime service protocol stack for low-power network effectively.

Contiki runs on a several different platforms, supports many microcontrollers and requires less memory.

***Network protocol architecture in Contiki***

Model TCP / IP traditional requires too much and memory resources as well as the ability to handle, it does not fit with the sensor nodes are more binding constraints. Therefore, the network protocol in Contiki has been redesigned by optimization, retaining only what is necessary based on the TCP / IP tradition. Protocol architecture consists of 4 floors (Figure 2.13)



*Figure 2.13: Network protocol architecture*

***Physical layer and Data Link***: This layer provides the physical elements of the network such as frequency, timing, voltage ... use many different standards and more specific standards in line with WSN network. The main task of this layer is selected frequency, the carrier frequency created, signaling, modulation and signal coding.

***Network layer:*** This layer decides routing capabilities, forwarding packets from node to node in the network. Here Contiki use protocol RPL, RPL using control messages (DIS, DIO, DAO) to set up the network topology and routing implementation.

***Transport layer:*** This layer 2 protocol is UDP and TCP, the first packet will be recalculated considering to ensure the validity and then use its port number to move to the application layer.

***Application Layer:*** This layer provides APIs for applications to use the services provided by the layer below. This floor applications such as HTTP, telnet ...

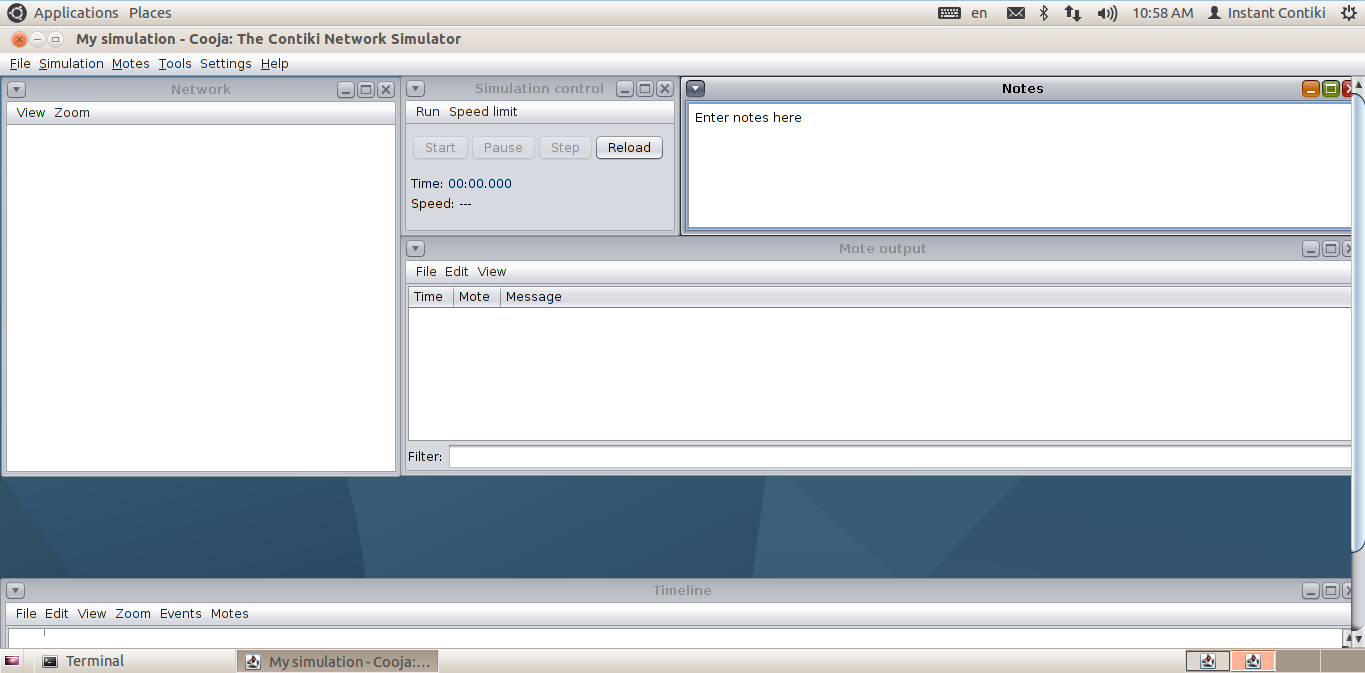
Contiki provides IP-based applications including IPv4 and IPv6 via dual stack communicate: uIP and Rime. The application can operate in one of two protocols, simultaneously on two protocols can operate or not based on any protocol. In addition, applications can uIP base activity and vice versa Rime, Rime supply applications can operate on the uIP.

* 1. **COOJA-SIMULATION TOOL**

Contiki provides this program to easily expand the application has been deployed, modify and improve on the open source operating system. It provides support library memory allocation and a list of related operations, as well as abstract concepts and mechanisms of action of low-power networks. It is a highly intuitive operating system because in addition to the test application it also contains some simulation help in understanding how it works. One of the Java-based plugin that is Cooja.

COOJA (Contiki OS Java) is a simulation tool for wireless sensor network is built on the Contiki operating system, is the first program written in Java but allows for compilation and simulation of sensor nodes language C. it supports the user in the basic operation of the buttons and the important thing is that the new device can completely be added by the user.

Cooja has simple interface (Figure 2.14) and easy to use, allows users to conduct simulated a sensor network. Users can easily perform tasks such as changing the parameters necessary to conduct simulations and evaluate the most effective way



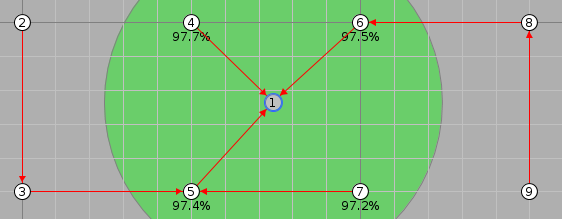
*Figure 2.14: Cooja screenshot*

# CHAPTER III: DEPLOY AND RESULTS

1. **DEPLOY**
   1. **NETWORK CONTRUCTION**
      1. **NETWORK TOPOLOGY**

In this project, when creating the network to run the simulation on Cooja, the mote was chosen as CoojaMotes. Network topology consists of nine nodes (Figure 3.16), where node 1 acts as a coordinator.

At the end of the initial network initialization process, the RPL tree is formed (Figure 3.1).



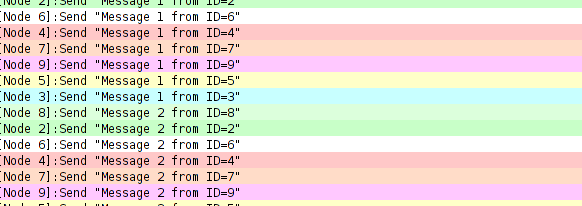
*Figure 3.1: Network Topology*

* + 1. **GATHERING DATA**

To represent the process of collecting data in the network, the nodes will create and send to the UDP packet coordinator the content of the payload: “Message num from ID=node\_id” where number is the id of the packet, node\_id is the id of node.

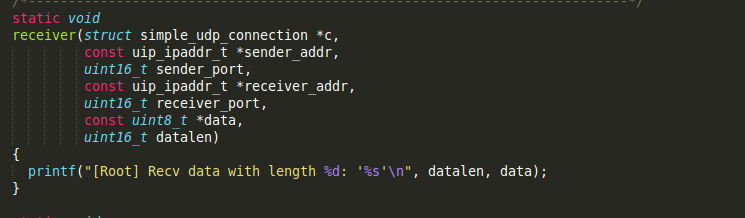
.

*Figure 3.2: List message coordinator receive.*

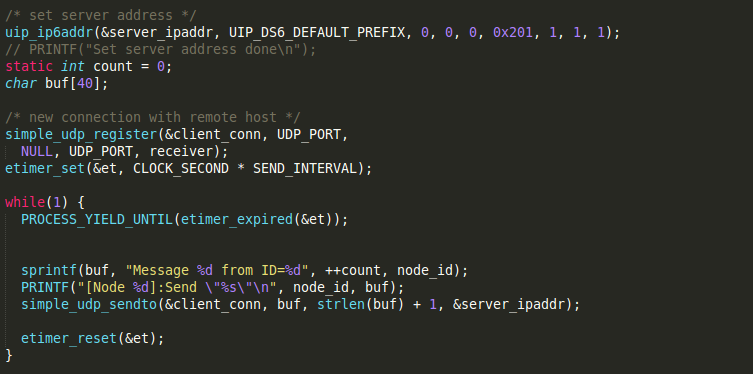
**

*Figure 3.3: List message receive from node*

This is snip code of coordinator and node for sending and receiving UDP packet.



*Firgure 3.4: Snip code for receiving UDP packet.*



*Firgure 3.5: Snip code for sending UDP packet.*

* + 1. **EVALUATE NETWORK PERFORMANCE AT DIFFERENT SERVICE LEVELS.**

In this project, there are two levels of QoS-0, QoS-1. For each level of service quality, combined with the change in the frequency of the nodes sending data packets, we measured a few parameters to assess the performance of the network in each case.

Three parameters are given:

* PDR (Packets Delivery Rate): Rate of received packet data successfully.
* Latency (ms): Average latency of data packets.
* Energy Consumption (%): the average time of the nodes on the total testing time.
  1. **LEVEL OF QUANLITY OF SERVICE**

1. **RESULTS**

Run the test at different service levels and change the frequency of sending data packets at the nodes, the results of measuring network parameters are shown in Table

|  |  |  |
| --- | --- | --- |
|  | Per 10s | |
| Qos 1 | Qos 0 |
| Latency |  |  |
| PDR |  |  |
| Energy Consumption |  |  |

|  |  |  |
| --- | --- | --- |
|  | Per 5s | |
| Qos 1 | Qos 0 |
| Latency |  |  |
| PDR |  |  |
| Energy Consumption |  |  |

|  |  |  |
| --- | --- | --- |
|  | Per 1s | |
| Qos 1 | Qos 0 |
| Latency |  |  |
| PDR |  |  |
| Energy Consumption |  |  |

# CHAPTER IV: CONCLUSTION

Providing quality service levels for low-power wireless sensor networks is essential, and dynamic control of service levels is the same. We cannot re-install a wireless sensor network just to change its latency to reduce energy consumption, which take a lot of time to deploy a low-power sensor network. From the analyzes and designs of the sensor network to data collection, with the results achieved, the project has completed its initial objectives including:

- Learn about wireless sensor networks.

- Learn about communication standards in wireless sensor networks.

- Introduces new architecture for wireless sensor networks using methods multiple access time slot and channel hopping (TSCH).

- Develop quality service levels for wireless sensor networks.

- Develop dynamic control mechanisms for service quality levels.

- Install and run the experiment on Cooja and take conclusions and reviews.

However, there are some limitations in implementation:

- The new test results are only made on the simulator not put into reality.

- The test has few nodes, because the broadcast control message has not yet been executed but only unicast from the coordinator to each node.

- The mobile gateway was not included in the test.

The construction of service levels in low-power wireless sensor networks has been completed, but a better control mechanism needs to be developed to allow for greater control of the networks. This is very difficult due to the death of the Ipv6 broadcast on the low-power wireless sensor network which is different from the IPv4 network is using, cannot send the broadcast indiscriminately will cause the death of the network due to the sensor button is very small, and requires long-term energy savings.