ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Informatica

Named Data Networking in IoT based sensor devices

Relatore:
Chiar.mo Prof.
Marco Di Felice

Correlatore:
Chiar.mo Prof.
Giovanni Pau

Presentata da:
Davide Aguiari

Sessione III
Anno Accademico 2015/2016

"Look behind you, a Three-Headed Monkey!"

Guybrush Threepwood

Introduction

In a world running on a "smart" vision, the **Internet of Things** (**IoT**) progress is going faster than ever. The term *"things"* is not just about computer, people and smartphone, but also sensors, refrigerators, vehicles, clothing, food and so on. Internet of Things is the possibility to provide an **IP address for every item**, so it will have an interface on the Internet network. The household devices will not just being commanded and monitored remotely then, but they will have an active main character role, establishing a communication network between them.

Another important cause of the development of Internet of Things is the **lower cost of the electronic components**: with only a few euros we are now able to buy devices which are more powerful and faster than ever, for example, *Arduino* and *Raspberry Pi*. These devices, if well configured, are able to evolve other household items born before this revolution, providing an IP address to most of the ordinary items.
Thinking about that, research and industry have started the implementation of projects in various sectors: domotics, medicine, sensors engineering, environment monitoring, etc.
This thesis aims to show how the IoT world, in conjunction with the research of techniques of environment pollution monitoring, could be a good path to follow; in the *LIP6* lab (*Laboratoire d'Informatique de Paris 6*) our team and I studied an innovative way to solve two different problems with the same tool:

- The pollution in the city of Paris, due to the particulate matter,
- Monitoring of fire in woods in remote areas of Thailand.

The thesis will begin describing a general overview, the state of art, of the IoT world and of sensors networks, checking its potential and any restrictions, if present. Then, every engineering aspect of the realized project, will been described in detail.
This thesis will also prove that nowadays we have the right items and components for the realization of reliable low-cost sensors.
The ultimate purpose is to verify the introduction of new network protocols like **NDN** (**Named Data Networking**) to evaluate their performances and efficiency. NDN is one of five projects funded by the *U.S. National Science Foundation* under its *Future Internet Architecture* Program.
In chapter 5 I will propose the simulations output obtained by *NS3* (*Network Simulator*): a Thai-like scenario simulation using *NDNSim* and *ChronoSync* application will be present.

Contents

List of Figures

Chapter 1

State of the art

Figure 1.1: Scott Bedford/Shutterstock

According to Cisco *Internet Business Solutions Group* (IBSG), the number of Internet devices will reach 50 billions by 2020.
The increase of new boards and smaller and smaller powerful and efficient sensors made the computer science take the big step.

This chapter aims to describe the typical scenario of these little components which are revolutionizing the technology, in an almost imperceptible way i.e. that of **Internet Of Things**.
Up next, a further examination of an aspect of the IoT world will be analyzed: **Wireless Sensor Networks** (**WSNs**), low-cost sensors typically adopted at industrial level or domestic level, to get environmental data.

1.1 IoT

1.1.1 Definition

The term *Internet of Things* was first coined by Kevin Ashton in the context of supply chain management in 1999. [2]
However, in the past decade, the definition has been more inclusive covering wide range of applications like healthcare, utilities, transport, etc. [3]
Even if the meaning of "*Thing*" changes over time, it remains the idea that is is the basis of M2M (Machine to Machine) communication: exchange data without any human intervention.
This is leading to a radical transformation of the Internet in a huge interconnected sensors network, which not only physically interact with the world or sense measures, but it supplies services to analyze data and to interact with them in real time.
Thanks to prevalence of devices enabled by open wireless technology such as *Bluetooth Low energy*, *Radio Frequency IDentification* (RFID), Wi-Fi 802.11ah, and many others as well as embedded sensor and actuator nodes, IoT has stepped out of its infancy and is on the verge of transforming the current static Internet into a fully integrated Future Internet [4].
Over the years, several definitions are assigned to IoT.
IEEE (Institute of Electrical and Electronic Engineers), in its special report on Internet of Things issued in March 2014, described the phrase "Internet of Things" as: *A network of items — each embedded with sensors — which are connected to the Internet.* [5]

Thanks to the transition from Ipv4 to Ipv6 which happened a few years ago, we have an incredible breakthrough. Every household device could potentially be equipped with an unique IP address, which can differentiate it from the others and provide to it an entry to communicate in the network.
The relationship between real objects and virtual objects are well described in this IoT definition:
"*The Internet of Things represents a vision in which the Internet extends into the real world embracing everyday objects. Physical items are no longer disconnected from the virtual world, but can be controlled remotely and can act as physical access points to Internet services. An Internet of Things makes computing truly ubiquitous.*" [6]

As identified by the authors in [7], Internet of Things can be realized in three paradigms:

- Internet-oriented (middleware)
- Things-oriented (sensors)
- Semantic-oriented (knowledge)

Finally, IoT could be seen as a "*Interconnection of sensing and actuating devices providing the ability to share information across platforms through a unified framework, developing a common operating picture for enabling innovative applications. This is achieved by seamless ubiquitous sensing, data analytics and information representation with Cloud computing as the unifying framework.*"[8]

1.1.2 Architecture

Internet of Things spreads over different levels: from the social level, through the application level, till the software architecture, as shown in the figure below:

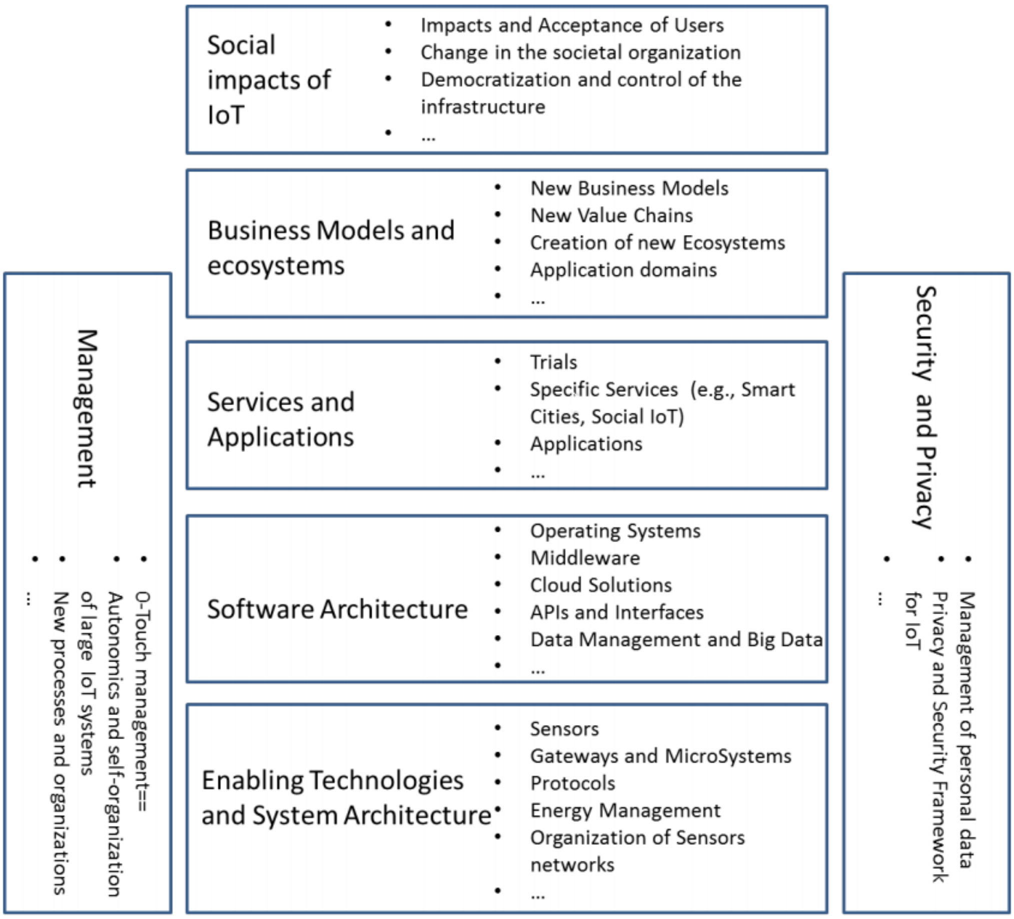


Figure 1.2: Technological and social aspects related to IoT [9]

Generally, Internet of Things does not configure itself in a specific area, but it is the result of different mechanisms that interact among them, from the software to each low-level components.

IoT operating systems are designed to run on **small-scale components in the most efficient way possible**, providing basic functionalities even if nowadays the boards are becoming increasingly powerful and tiny.

Self-management and **self-optimization** of each individual component and/or subsystem maybe strong requirements.

In other words, autonomous behaviors could become the norm in large and complex IoT systems. [9]

Let's analyze now the different key points of the IoT world:

- User(s)
- Physical entity
- Device
- Sensor OS

User

A physical person or an active digital entity, a service, that has a **goal**. This goal is achieved via interaction with the physical environment. This interaction is mediated by the IoT.
The user can be active when it modifies the environment or passive when it only retrieves values from it.

Physical entity

The Physical Entity is an **identifiable part** of the physical environment that is of interest to the User for the completion of his goal. [10] Physical Entities can be almost any object or environment; from humans or animals to cars; from store or logistics chain items to computers; from electronic appliances to jewelery or clothes.

Device

A device combines the users to the physical entities, simply **suppling data** or interfaces among them when they're close to the entities.
From a functional point of view, devices can belong to any of the following types:

- **Tags** – Passive sensors that are able to be read from "readers" in order to set a unique identifier to the device.

RFID is a perfect solution for providing this unique identification of "things." The tag of an RFID is used to carry data, which is located on the object to be identified.
This consists of a coupling element, such as an antenna, and an electronic microchip, less than one-third millimeter in size.

- **Sensors** provide information about the physical entity they monitor. Information in this context ranges from the identity of the physical entity to measures of the physical state of the physical entity.

- **An interrogator or reader** reads the transmitted data. Compared with tags, readers are larger, more expensive and power-hungry. They usually transmits a low-power radio signal to power the tag.
The tag then selectively reflects energy and thus transmits some data back to the reader, communicating its identity, location and any other relevant information.
Frequencies currently used for data transmission by RFID typically include 125 kHz (low frequency), 13.56 MHz (high frequency) or 800-960 MHz (ultra-high frequency).

- **Actuators** can modify the physical state of a physical entity. Actuators can move (translate, rotate, etc.) simple physical entities or activate/deactivate functionalities of more complex ones.

Sensor Operating Systems

Most operating systems (OS) that may be used for IoT were designed for wireless sensor networks (WSN) like *RIOT*, *TinyOS* and *Contiki*.
OSs for sensor nodes follow either one of two different design concepts [12]:

- **Event-driven systems**, where every action an OS has to perform is triggered by an event like a timer, an interruption indicating new sensor readings or an incoming radio packet;

- **Multi-threaded system**, where the OS multiplexes execution time between the different tasks, implemented as threads.

RIOT is one the main and promising operating system IoT-oriented.
The developers of RIOT claim that an IoT OS needs to fulfill the following requirements [11]:

- Minimal requirements to memory (RAM and program memory) and computing power
- Ability to run on constrained hardware without more advanced components like a memory management unit (MMU) or a floating-point unit (FPU)
- Standard programming interface
- Support for high-level programming languages
- Support to a variety of hardware platforms
- High degree of energy efficiency
- An adaptive and modular network stack
- Reliability

The strength of RIOT, and of the other IoT-oriented OS, is the low-power consuming: Linux, which is developer-friendly and well supplied with every library or driver you may need, lacks some of the minimum requirements above, as the CPU management in a low-power scenario, by comparison.
These points are illustrated in the figure below, which compares attributes of the cited OSs.

OS	Min RAM	Min ROM	MCU w/o MMU	16-bit processors
Contiki	<2kB	<30kB	✓	✓
Tiny OS	<1kB	<4kB	✓	✓
Linux	~1MB	~1MB	✗	○
RIOT	~1.5kB	~5kB	✓	✓

Figure 1.3: Hardware requirements for Contiki, TinyOS, Linux, and RIOT.
(✓) full support, (○) partial support,(✗) no support.

RIOT, and the like, are less supported by huge communities and they may have a hard learning curve. For example TinyOS neither provides C/C++ support.

OS	C Support	C++ Support	Multi-Threading	Modularity	Real-Time
Contiki	○	✗	○	○	○
Tiny OS	✗	✗	○	✗	✗
Linux	✓	✓	✓	✓	○
RIOT	✓	✓	✓	✓	✓

Figure 1.4: Key characteristics of Contiki, TinyOS, Linux, and RIOT.
(✓) full support, (○) partial support, (✗) no support.

Summarizing, we can identify the following key system-level features that Internet-of-Things needs to support [13]:

- **Devices heterogeneity**. IoT is characterized by a large heterogeneity in terms of devices taking part in the system, which are expected to present very different capabilities from the computational and communication standpoints.

- **Ubiquitous** data exchange through proximity wireless technologies.
  Wireless communications technologies enables smart objects to become networked.
  The ubiquitous adoption of the wireless medium for exchanging data also pushes towards the adoption of cognitive/dynamic radio systems in order to overcome issues in terms of spectrum availability [14].

- **Scalability**. As everyday objects get connected to a global information infrastructure, scalability issues arise at different levels, including:

  – naming and addressing – due to the sheer size of the resulting system, even if IPv6 and NDN solve the problem;

  - data communication and networking – due to the high level of interconnection among a large number of entities, pushing new solutions in terms of spectrum usage;
  - information and knowledge management – due to the possibility of building a digital counterpart to any entity and/or phenomena in the physical realm;
  - service provisioning and management – due to the massive number of services/service execution options that could be available and the need to handle heterogeneous resources;

- **Energy-optimized solutions**. In IoT minimizing the energy to be spent for communication/computing purposes is primary constraint.

- **Self-organization capabilities**. The complexity and dynamics that many IoT scenarios presents calls for distributing intelligence in the system, making smart objects able to autonomously react to a wide range of different situations, in order to minimize human intervention. In this field, the artificial intelligence brings a huge effort in order to make the device self-configurable. [15].

- **Localization and tracking capabilities**. As entities in IoT can be identified and are provided with short-range wireless communications capabilities, it becomes possible to track the location (and the movement) of smart objects in the physical realm.

- **Embedded security and privacy-preserving mechanisms**. Due to the tight entanglement with the physical realm, IoT technology should be secure and privacy-preserving by design. More details below.

1.1.3 Security

Data security and privacy will play an important role in IoT deployments. The IoT world is **tightly connected to the personal informations**; so the data

security and privacy are a critical factor would need attention.
To protect data against the first type of attack, memory is protected in most tag technologies and solutions have been proposed for wireless sensor networks as well. For example, tags protect both read and write operations on their memory with a password [16].
Another way to protect data against attacks is protecting messaging according to the *Keyed-Hash Message Authentication Code* (HMAC) scheme: this is based on a common secret key shared between the tag and the destination of the message, which is used in combination with a hash function to provide authentication. [18]
All the solutions proposed to support security use some **cryptographic methodologies**, even if they are power and energy hungry. Such solutions cannot be applied to the IoT, given that they will include elements (like RFID tags and sensor nodes) that are seriously constrained in terms of energy, communications, and computation capabilities [17].

Data confidentiality

In IoT scenarios, a fundamental issue is the data confidentiality. It is mandatory that **only authorized entities can access and modify data**.
This is particularly relevant in the business context, whereby data may represent an asset to be protected to safeguard competitiveness and market values.
A transparent access control mechanism and a strong authentication process can achieve this goal. [19].

Privacy

The concept of privacy is deeply rooted into our civilizations; the ways in which data collection, mining, and provisioning will be accomplished in the IoT are completely different from those that we now know and there will be an amazing number of occasions for personal data to be collected.
Furthermore, the cost of information storage continues to decrease and is now approaching 10^{-9} euro per byte; it means that once information is generated, will most probably be retained indefinitely, which should push governments to legislate

on private daya storage. The data described above, **should be stored only until it is strictly needed**. [7]
In early-stage IoT deployments security solutions have mostly been devised in an ad hoc way.
A recent report by *Hewlett Packard Enterprise's Aruba* division confirms that 84% of organizations analyzed have experienced an IoT-related security breach. [41]
In the perspective of an open IoT eco-system, whereby different actors may be involved in a given application scenario, a number of security challenges do arise [20].

1.2 Wireless Sensor Networks (WSN)

Figure 1.5: Wireless Sensor Networks [49]

1.2.1 Definition

Sensor networks are **dense wireless networks** of small, low-cost sensors, which collect and disseminate environmental data. They facilitate monitoring and

controlling of physical environments from remote locations with better accuracy. They have applications in a variety of fields such as environmental monitoring, indoor climate control, surveillance, structural monitoring, medical diagnostics, disaster management, emergency response, ambient air monitoring and gathering sensing information in inhospitable locations [21].
Sensor nodes have various **energy and computational constraints** because of their inexpensive nature and ad-hoc method of deployment.
Considerable research has been focused at overcoming these deficiencies through more energy efficient routing, localization algorithms and system design [22].

1.2.2 Architecture

This small device, called sensor node, consists of sensor, wireless communication device, small microcontroller and energy source.
Wireless sensor network has some unique characteristics such as large scale of deployment, mobility of nodes, node failures, communication failures and dynamic network topology. In addition, each sensor node has constraints on resource such as energy, memory, computation speed and bandwidth because of the constraints on size and cost.

Many applications of the WSN require secure communications. However, wireless sensor networks are prone to different types of malicious attacks, such as impersonating, masquerading, interception for misleading because of the wireless connectivity, the absence of the physical protection and the unattended deployment [23].

The main WSN components are the **transducers**.
A transducer is a device that converts energy from one domain to another.
It converts the quantity to be sensed into a useful signal that can be directly measured and processed. Since much signal conditioning and digital signal processing is carried out by electronic circuits, the outputs of transducers that are useful for sensor networks are generally voltages or currents [24].

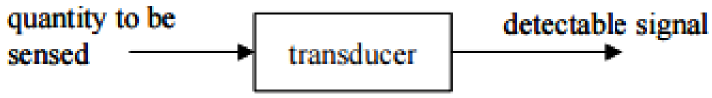


Figure 1.6: Sensory Transducer scheme [24]

Generally, a sensor node is a tiny device that includes three basic components:

1. a sensing subsystem for data acquisition from the physical surrounding environment
2. a processing subsystem for local data processing and storage,
3. a wireless communication subsystem for data transmission.

In addition, a power source supplies the energy needed by the device to perform the programmed task. This power source often consists of a battery with a limited energy budget.
There are different sensors such as pressure, accelerometer, camera, thermal, microphone, etc. They monitor conditions at different locations, such as temperature, humidity, vehicular movement, lightning condition, pressure, soil makeup, noise levels, the presence or absence of certain kinds of objects, mechanical stress levels on attached objects, the current characteristics such as speed, direction and size of an object. [25]

A sensor node typically consists of five main parts [26]:

1. One or more **sensors** gather data from the environment.
2. The central unit in the form of a **microprocessor** manages the tasks.
3. A **transceiver** communicates with the environment
4. A **memory** is used to store temporary data or data generated during processing.

5. The **battery** supplies all parts with energy. To assure a sufficiently long network lifetime, energy efficiency in all parts of the network is crucial.

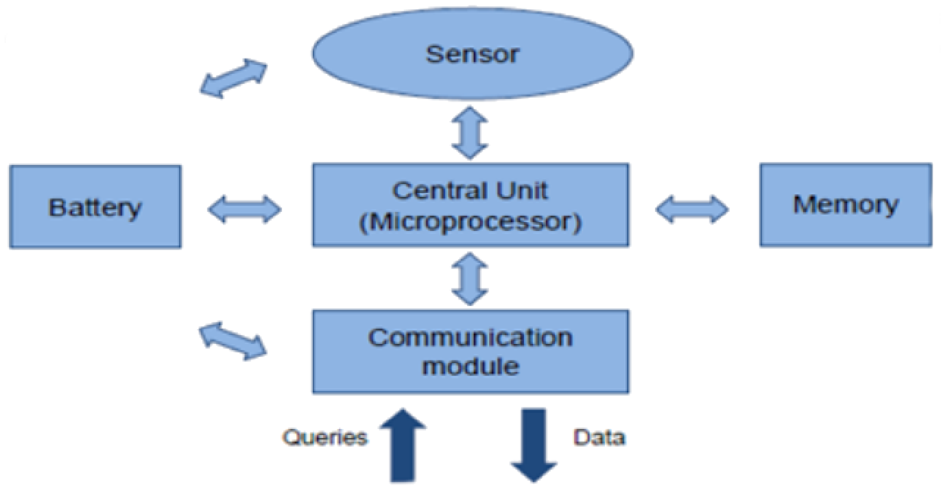


Figure 1.7: Architecture of a smart sensor node [42]

The development of sensor nodes is influenced by increasing device complexity on microchips, high performance, wireless networking technologies, a combination of digital signal processing and sensor data acquisition, advances in the development of microelectromechanical systems, and availability of high performance development tools [27].

1.2.3 Characteristics of WSN

The main characteristics of a WSN include [28]:

- Power consumption constrains for nodes using batteries or energy harvesting
- Ability to cope with node failures
- Mobility of nodes
- Dynamic network topology

- Communication failures
- Heterogeneity of nodes
- Scalability to large scale of deployment
- Ability to withstand harsh environmental conditions
- Ease of use
- Unattended operation
- Power consumption

1.2.4 Fields of applications of wireless sensor network

1. **Security and surveillance**: the sensors could be used in order to monitor private properties through cameras or drones.

2. **Environmental Monitoring**: The term *"Environmental Sensor Networks"* has evolved to cover many applications of WSNs to earth science research. This includes sensing volcanoes, oceans, glaciers, forests, etc.

    - *Air pollution monitoring*: Wireless sensor networks have been deployed in several cities to monitor the concentration of dangerous gases for citizens. These can take advantage of the ad-hoc wireless links rather than wired installations, which also make them more mobile for testing readings in different areas.
    - *Forest fires detection*: A network of Sensor Nodes can be installed in a forest to detect when a fire has started. The nodes can be equipped with sensors to measure temperature, humidity and gases which are produced by fires in the trees or vegetation. The early detection is crucial for a successful action of the firefighters; thanks to Wireless Sensor Networks, the fire brigade would be able to know when a fire is started and how it is spreading [25].

3. **Health Applications**: In some modern hospital sensor networks are constructed to monitor patient physiological data, to control the drug administration track and monitor patients and doctors.

4. **Energy Control System**: the sensors are able to check the flow current in high voltage wires or control the train switch points.

5. **Agriculture Applications**: Wireless sensors free the farmer from the maintenance of wiring in a difficult environment. Wireless sensor networks are also used to control the temperature and humidity levels inside commercial greenhouses.

6. **Industrial applications**: the sensors in this field are used for a wide range of purposes, from the production control to machine automation.

7. **Structural monitoring**: Wireless sensors can be used to monitor the movement within buildings and infrastructure such as bridges, flyovers, embankments, tunnels etc

1.2.5 WSN Based Air Pollution Monitoring Systems

Air pollution in urban areas with ubiquitous emission sources attracts extensive attentions worldwide due to the tremendous impacts on human lives at anytime and anywhere.
Data acquired by these stations can be utilized for building pollution maps and models that provide authorized environmental situation information and prediction [30].

According to this study [29] the existing WSN based air pollution monitoring systems are classified into three categories based on the carriers of the sensor nodes:

1. **Static Sensor Network (SSN)**: sensor nodes are usually mounted on the streetlight or traffic light poles, or carefully selected locations.
   In [47] a study was conducted in a similar backgrounds as ours: an outdoor

ambient real-time air quality monitoring system was implemented and tested. The sensors used get the concentration of O_3, NO_2, CO and H_2S and GPRS location over 1-minute sampling. The authors also built a Web and mobile App to view the data. A solar panel was utilized to solve the power constraint issue of the sensor nodes.

2. **Community Sensor Network (CSN)**: sensor nodes are carried by the public communities, usually by volunteers or people who are keen on air quality.
   [48] introduces a interesting project in a CSN scenario: an outdoor air quality sensing system (*P-Sense*) based on the participatory sensing technology. The sensors used collect multiple values like CO_2, CO, VOCs, H_2, temperature and relative humidity and they exchange them via Bluetooth. Data are acquired by the sensors and transmitted to the smartphone. Even in this case, the authors developed a Web app to show the results.

3. **Vehicle Sensor Network (VSN)**: Sensor nodes are carried by the public transportations or specially equipped cars.

Chapter 2

NDN

2.1 Definition

The NDN project transforms the concept of host-centric network architecture (IP) to a data-centric network architecture and is one instance of a more general network research direction called **Information-Centric Networking** (ICN).
IP was designed to create a communication network, where packets identified only communication endpoints. Sustained growth in e-commerce, digital media, social networking, and smartphone applications have led to dominant use of the Internet as a distribution network. [34]
NDN aims to generalize the IP's thin waist, such that **packets can name objects other than endpoints in a communication**.

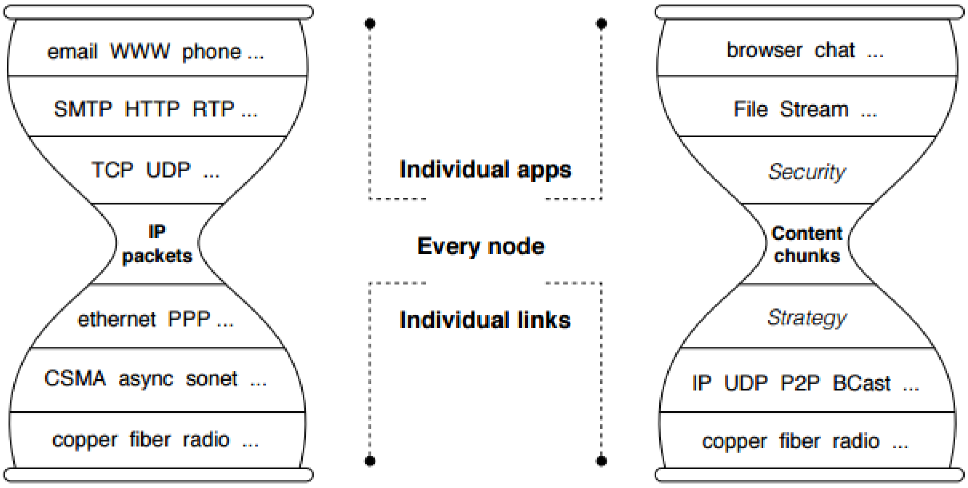


Figure 2.1: CCN moves the universal component of the network stack from IP to chunks of named content [35]

NDN changes the semantics of network service from delivering the packet to a *given destination address to fetching data identified by a given name.*
A name in a NDN packets has a completely different meaning: it could be a book's chapter, a command in a system, a sensor data, a frame in a video etc.

2.2 Architecture

Coumminication in NDN is driven by two kinds of packets:

- **Interest Packet**: A consumer puts the name of a desired piece of data into an Interest packet and sends it to the network. Routers use this name to forward the Interest toward the data producer(s).

- **Data Packet**: Once the Interest reaches a node that has the requested data, the node will return a Data packet that contains both the name and the content, together with a signature by the producer's key.

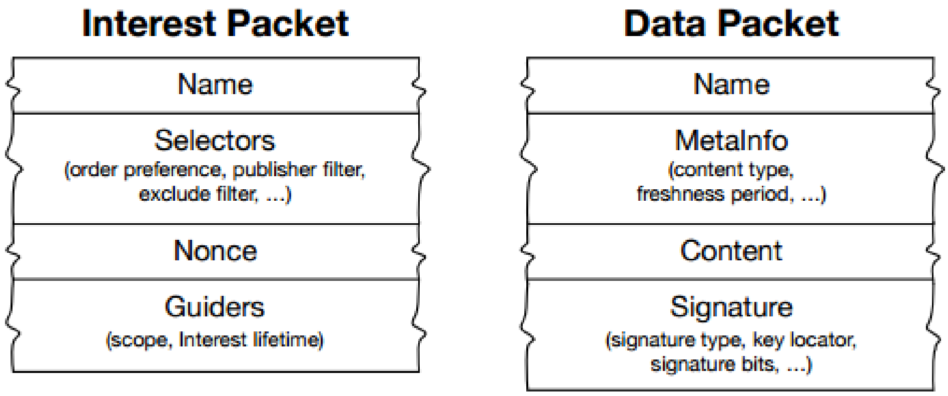


Figure 2.2: CCN packet types [35]

In order to carry out the Interest/Data packet, NDN routers use a *Forwarding Strategy* with three data structures:

- **Forwarding Information Base** (FIB): populated by a name-prefix based routing protocol and multiple output interfaces per prefix

- **Pending Interest Table** (PIT): The table contains different entries; each entry records the Data name with its incoming and outgoing interfaces.

- **Contest Store** (CS): The Content Store is a cache of Data packets received. When an Interest packet comes, an NDN router checks the Content Store first for matching data; if it exists the router returns the Data packet on the interface from which the Interest came, else it looks up the name in its PIT.

In the absence of a matching PIT entry, the NDN router will forward the Interest toward the producer based on information in the FIB according to the Forwarding Strategy that may decide to drop it in certain situations: in this way it avoids DoS attacks or links congestion.

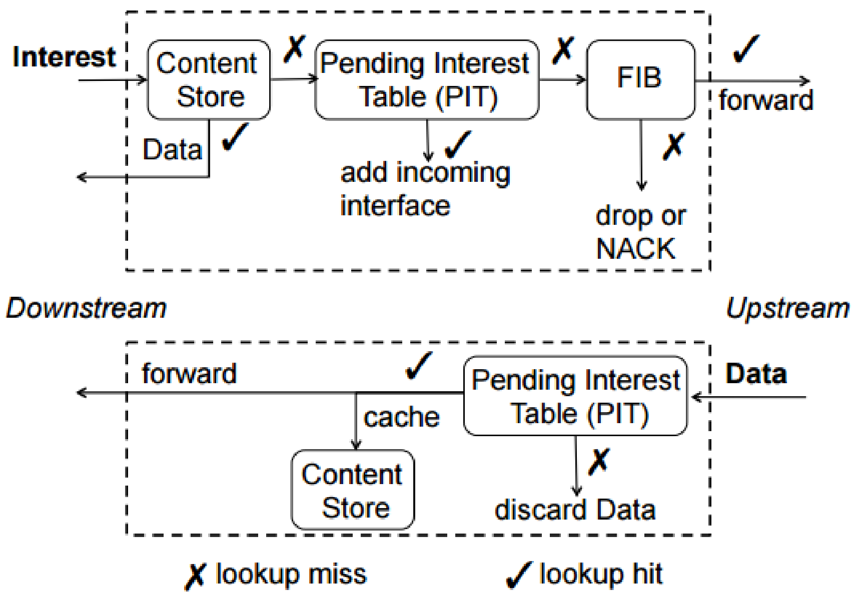


Figure 2.3: Forwarding Process at an NDN Node [34]

When a Data packet arrives, an NDN router finds the matching PIT entry, forwards the data to ALL downstream interfaces listed in that PIT entry and it then removes that PIT entry and caches the Data in the Content Store.
Data packets always take the reverse path of Interests and one Interest packet results in one Data packet on each link, providing flow balance.

Neither Interest nor Data packets carry any host or interface addresses; routers forward Interest packets toward data producers based on the names carried in the packets, and forward Data packets to consumers based on the PIT state information set up by the Interests at each hop [34].
The need of any source or destination notion becomes superfluous.

2.3 Names

NDN names are *opaque* to the network. This means that each application choose the naming scheme that fits its needs.
Tha Names structure is hierarchical and every parts of the name is divided by '/'.

For example:

/upmc/lip6/106/temperature

may retrieve the temperature in the office 106 into the LIP6 lab in the UPMC University.
Interest selectors in conjunction with longest prefix matching usually get the desired data through one or more iterations.
Data that may be retrieved globally must have globally unique names, but names used for local communications may require only local routing (or local broadcast) to find matching data.

2.4 Security

TCP/IP leaves responsibility for security to the endpoints while NDN **secures the data itself** by requiring data producers to **cryptographically sign** every Data packet. [35] This inherently solves the lack of security in a IoT scenario.
It also supports finegrained trust, allowing consumers to reason about whether a public key owner is an acceptable publisher for a data.
Applications can control access to data via encryption and distribute keys as encrypted NDN data. NDN's use of multipath forwarding, together with the adaptive forwarding strategy module, mitigates prefix hijacking because routers can detect anomalies caused by hijacks and retrieve data through alternate paths. [36]
Since NDN packets reference content rather than devices, it is harder to maliciously target a particular device, although mitigation mechanisms will be needed against other NDN-specific attacks, e.g., Interest flooding DoS [37].

Chapter 3

The project

Environmental pollution due to vehicular traffic, coal plants and waste incineration is a serious danger which could undermine the ecosystem, and is one of the major cause of deaths per year.
According to a recent survey of the *Stockholm Environment Institute* (SEI), at least 3.4 millions of premature births are caused by air pollution. It is also estimated that more than 90% of the world population lives in places where air pollution exceeds safe limits set by the *World Health Organization* (WHO) [38].
China has the worst record, a place where pollution kills more than 1 million people per year, followed by India (600.000) and Russia (140.000). Italy is the third most polluted country of European Union, with more than 12.000 deaths per year.

The project aims to realize a low-powered, low-cost single-board computer that is able to constantly monitor different measures as:

- Particulate matter 10 (PM_{10})
- Particulate matter 1.0 ($PM_{1.0}$)
- Particulate matter 2.5 ($PM_{2.5}$)
- Nitrogen dioxide (NO_2)
- Formaldehyde
- Temperature

- Air pressure
- Relative humidity

The different values, geo-localized via GPS, are measured in a time interval and then sent to a centralized database to provide important data to organization for pollution monitoring.
If, until now, the lack of appropriate or very expensive tools could have compromised a large-scale monitoring, this project now lays the groundwork for the construction of low-cost tools that can give reliable measurements without needing of professional equipments.

3.1 Objectives and requirements

The project sets some fundamental requirements as:

1. **Low cost components**: to allow a great scale production, it is necessary being competitive in a market where there are industrial sensors, which are functional but also expensive;
2. **Low energy consumption**, to minimize the necessity of recharge the board;
3. **Certified sensor measurement**: to ensure a reliable reading of the atmospheric values;
4. **Constant connection**: to have a real-time feedback of the measurement

3.2 Context

The particulate matter is the set of substances suspended in the air and it is a mix of: fibers, metals, solid and liquid pollutants, silica and carbonaceous particles.
In urban areas, the most dangerous pollutant is caused both by natural phenomena

as dust, fires or erosion and anthropogenic factors as combustion engines, domestic heating, mechanical and physical processing, incinerators, power plants and tobacco smoke. [39]
The total dust emissions can be broken down depens on their particles dimensions. In details there are three different categories:

- PM_{10} – particulate is made up of particles sizes of less than 10μm (less than a hundredth of a millimeter), is an inhalable dust, i.e. it enters the nose and the larynx. The particles between 2.5 and 5μm ish are deposited into the bronchioles earlier. It remains in the air for 12 hours.

- $PM_{2,5}$ – fine particulate with less than 2.5μm (a quarter of a hundredth of a millimeter) in diameter, is a thoracic dust, i.e. it is able to penetrate deeply into the lung.

- $PM_{1.0}$, with less than 1μm in diameter, breathable dust able to penetrate deeply into lung and alveoli. It can remains in the air up to a month.

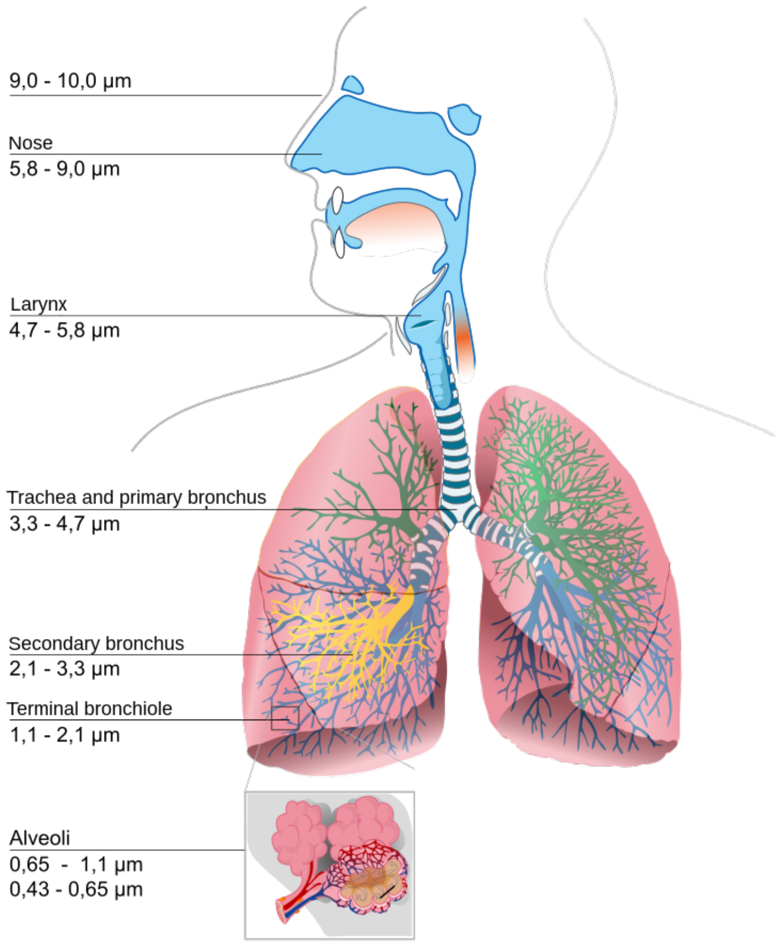


Figure 3.1: The penetration of particulate in the respiratory system [43]

The Italian legislation in the field of pollution (D.Lgs.155/2010) governs only the presence of PM_{10} and $PM_{2.5}$ in the air:

- PM_{10}: The maximum annual limit level is $40\mu g/m^3$; the daily level is $50\mu g/m^3$, not to be exceeded more than 35 times a year.

- $PM_{2.5}$: The limit level is $25\mu g/m^3$.

AQI	Air Pollution Level	Health Implications	Cautionary Statement (for PM2.5)
0 - 50	Good	Air quality is considered satisfactory, and air pollution poses little or no risk	None
51 -100	Moderate	Air quality is acceptable; however, for some pollutants there may be a moderate health concern for a very small number of people who are unusually sensitive to air pollution.	Active children and adults, and people with respiratory disease, such as asthma, should limit prolonged outdoor exertion.
101-150	Unhealthy for Sensitive Groups	Members of sensitive groups may experience health effects. The general public is not likely to be affected.	Active children and adults, and people with respiratory disease, such as asthma, should limit prolonged outdoor exertion.
151-200	Unhealthy	Everyone may begin to experience health effects; members of sensitive groups may experience more serious health effects	Active children and adults, and people with respiratory disease, such as asthma, should avoid prolonged outdoor exertion; everyone else, especially children, should limit prolonged outdoor exertion
201-300	Very Unhealthy	Health warnings of emergency conditions. The entire population is more likely to be affected.	Active children and adults, and people with respiratory disease, such as asthma, should avoid all outdoor exertion; everyone else, especially children, should limit outdoor exertion.
300+	Hazardous	Health alert: everyone may experience more serious health effects	Everyone should avoid all outdoor exertion

Figure 3.2: World Air Quality Index [44]

3.2.1 Paris

Figure 3.3: Paris Air Quality [44]

Paris, French capital, is the most populated city in Europe and it encompassing a population of over 11 million people. In December 2016, the value of particulate matter in the city peaked the highest level in the last 10 years; it made the administration set new rules about vehicles circulation and emissions.

A project in the field of particulate monitoring perfectly suits into this critical situation.

The project in Paris requires the **single-boards computer setup into backpacks** issued to first-grade children, in order to constantly monitor the pollution along the home/school path.

In February 2017, the sensors developed have passed the preliminary tests successfully at the LSCE labs (*Laboratoire des Sciences du Climat et de l'Environnement*) in Saclay and they're being further tested at *AirParif* (*Association de surveillance de la qualité de l'air*), Paris.

Figure 3.4: AirParif particulate matter test, February 2017

In this case, this WSN scenario belongs to the CSN (*Community Sensor Network*).
In CSN (or Participatory Sensing) systems, the sensor nodes are typically carried by the users. They send data whenever a *FreeWifi* network is available. [32]
All the air pollution information is available to the public through the web page or mobile App.
According to [29] there are some advantages and disadvantages:
Advantages

1. Coupled data generators and consumers: local or personal air pollution information is available.

2. Public-driven property: the cost of the sensor board and the data transmission can be apportioned by the users.

3. Automatic gathering property: the sensor nodes are densely distributed at locations with gathering people automatically. Data with higher spatial resolution and accuracy are achievable in such case.

4. Mobility of sensor nodes: the mobility of the users enlarges the sensor node's geographic coverage.

5. Public behaviors acquisition ability: information such as the public movement patterns, and interaction between air quality and public behaviors, is achievable.

6. Cost efficiency: the sensor node utilizes low-cost components.

Disadvantages

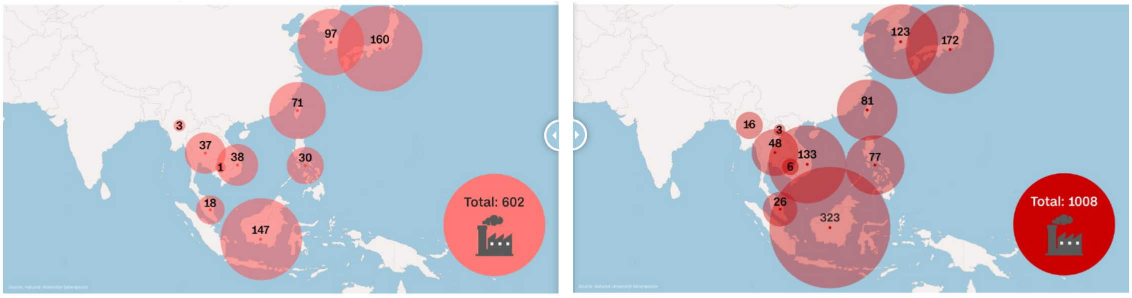
1. Low data accuracy and reliability: the sensor nodes are put into backpacks. Also, the users spend significant amount of time indoor or inside cars [33].

2. Privacy issues: the users may not want to make their location information public for privacy issues.

3. Serious constraint on energy consumption: the sensor nodes is powered by battery with small capacity.

4. Uncontrolled or semi-controlled mobility: the routes of the sensor nodes or users are pre-determined. The sensor nodes may squeeze into a small place with crowded people and cause redundant sampling. Some locations may never be visited.

5. Serious limitations on weight and size: the sensor node should be portable, which affects the accuracy, reliability and number of sensors equipped, because it is carried by user.



(a) Number of coal-fired plants in 2011 (b) Number of coal-fired plants by 2030

Figure 3.5: Number of coal-fired plants in Asia

3.2.2 Thailand

Coal emissions in Southeast Asia are projected to triple by 2030 (Figure 3.5). "*Air pollution in China and India has received a lot of scientific attention*" said Harvard University's Shannon Koplitz, a lead researcher in the Asian air pollution project. However, she says the "*impacts of planned coal power expansion in the rest of the Southeast and East Asian region have been understudied*" [40].
Koplitz ends by saying: "*Reliance on coal in emerging Southeast Asian countries will have substantial and long-lasting impacts on air quality and public health*".

Our project, in collaboration with the AIT (*Asian Institute of Technology*) in Bangkok, aims to monitor the pollution in the Chiang Rai area, in the north of Thailand.
Installing our sensors in specific points in the region, will enable not only to monitor the area, but also to **identify forest fires or villages fires on time**.
The sensors setup is scheduled for Spring 2017.

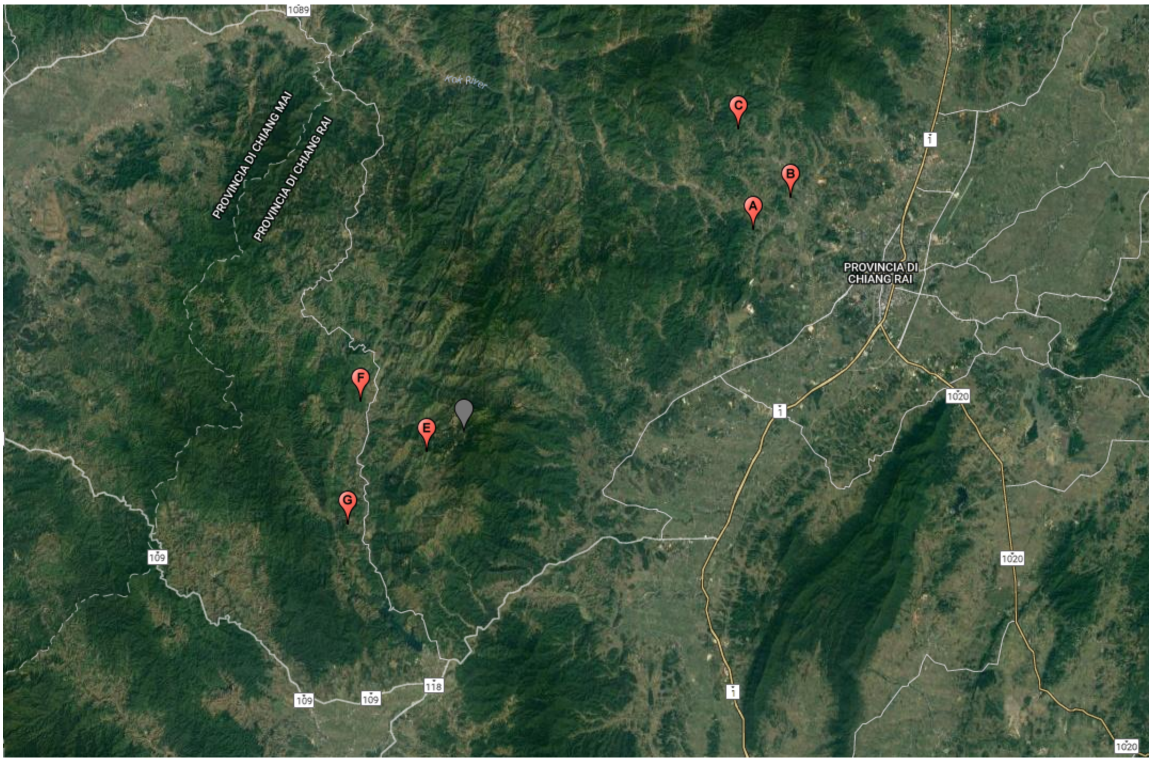


Figure 3.6: Red pins on our future monitoring stations in Chiang Rai area

In this case, this WSN scenario belongs to the SSN (*Static Sensor Network*). In SSN systems, the sensor nodes are typically mounted on the streetlight or traffic light poles, or buildings.
By utilizing the low-cost ambient sensors, the number of sensor nodes in SSN systems is much larger than that in the conventional monitoring systems.
As in the previous case, all the data is available to the public through the web page and mobile App.
According to [29] there are some advantages and disadvantages:

Advantages

1. Loose constraint on energy consumption: the sensor nodes are powered by batteries with large capacity or energy harvest devices or power line. In our case we are planning to use solar panels.

2. Multiple sensors per node: one sensor node can equip with several types of sensors because of the loose limitations on weight and size.

3. Accurate and reliable data: sensor node can integrate with assisting tools because of the loose limitations on weight and size.

4. Guaranteed network connectivity: once the stationary sensor node joined the network, the topology is fixed and the connectivity is guaranteed even if it has to be tested in the forest and with our distances between nodes.

5. Well calibrated and maintained sensors: the sensor nodes can be well calibrated and maintained by the professionals periodically.

Disadvantages

1. Careful placement of sensor nodes requirement: this is because of the location dependence of air pollutants. In our case this has been already done in the past.

2. Large number of sensor nodes requirement: data with sufficient geographic coverage and spatial resolution are only achievable by increasing the number of the stationary sensor nodes.

3. Customized network requirement: a customized wireless or wired network is required when the cellular network is not utilized.

3.3 Architecture

The project had a series of different boards in order to find the right balance between functionalities and requirements.
The prerequisites are:

- UART/I2C ports for the sensors
- Wifi 802.11 connection
- WPA2 Enterprise support for *eduroam* network
- EAP-SIM for *freeWifi* network

- Linux, in order to be more flexible
- Arduino supported
- Ethernet connection
- A microprocessor for real-time sampling
- Low energy consumption

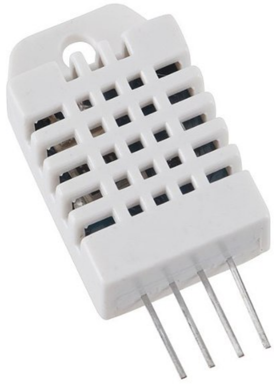
3.3.1 Sensors and components

The tested sensors chosen are:

- **AM2302 (DHT22) - Temperature and Relative Humidity**

The DHT22 is a basic, low-cost digital temperature and relative humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin.

- Low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 0-100% humidity readings with 2-5% accuracy
- Good for -40 to 80°C temperature readings ±0.5°C accuracy
- No more than 0.5 Hz sampling rate (once every 2 seconds)
- Body size 27mm x 59mm x 13.5mm (1.05" x 2.32" x 0.53")
- 4 pins, 0.1" spacing
- Weight: 2.4g



- **BMP180 - Barometric Pressure**

This precision sensor from Bosch is a low-cost sensing solution for measuring barometric pressure and temperature. The sensor is soldered onto a PCB with a 3.3V regulator, I^2C level shifter and pull-up resistors on the I^2C pins.

- Low cost
- Vin: 3 to 5VDC
- Logic: 3 to 5V compliant
- Pressure sensing range: 300-1100 hPa (9000m to -500m above sea level)
- Up to 0.03hPa / 0.25m resolution
- -40 to +85°C operational range, +-2°C temperature accuracy
- This board/chip uses I2C 7-bit address 0x77.

- **PM SENSOR**

The optical analyzer utilized, exploits the interaction between the ambient PM and a laser. These analyzers are small, lightweight and battery operated.
Base on the optical principle, the optical analyzers can be classified into three categories:

- direct imaging analyzers
- light scatting analyzers
- light obscuration (nephalometer) analyzers.

The sensor used is a light scattering optical analyzer.
This category of optical analyzers uses a high-energy laser as the light source.
When a particle passes through the detection chamber that only allows single particle sampling, the laser light is scattered by the particle. A photo detector detects the scatting light. By analyzing the intensity of the scatting light, researchers can deduce the size of the particle.
Also, the number of particle counts can be deduced by counting the number of

detecting light on the photo detector.

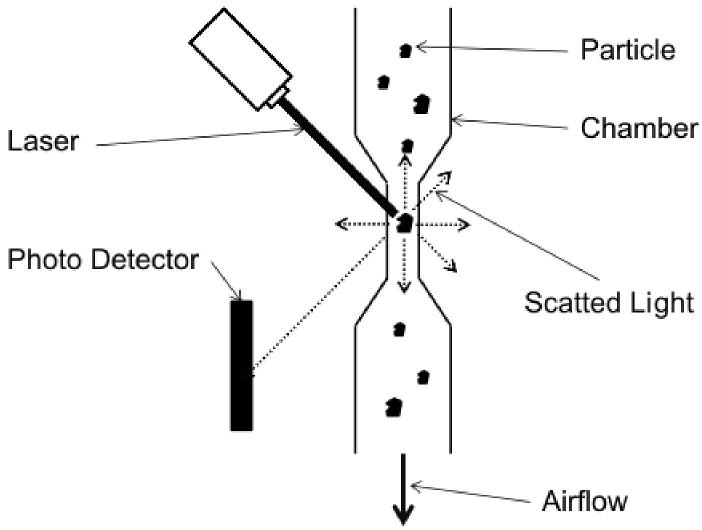


Figure 3.7: Basic Light Scatting Particle Counter [29]

The advantage of this approach is that a single analyzer can detect particles with different diameters simultaneously (i.e. $PM_{2.5}$, $PM_{1.0}$ and PM_{10}).
However, the particle counts need to be converted to mass concentration by calculation (depends on the particle counts, particle types and particle shapes) and this could introduce errors that further affect the precision and accuracy of the analyzers [29].

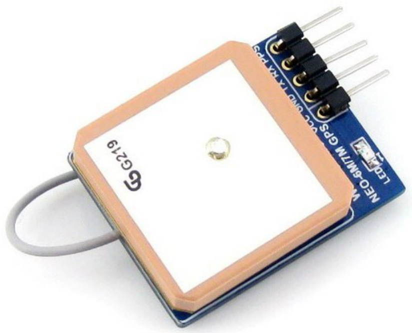
There are some other technical components like:

- **RTC DS3231 - Real time clock**
  The DS3231 is a low-cost, extremely accurate I^2C real-time clock (RTC) with an integrated temperature-compensated crystal oscillator (TCXO) and crystal.
  The device incorporates a battery input, and maintains accurate timekeeping when main power to the device is interrupted.

- Real-Time Clock Counts Seconds, Minutes, Hours, Date of the Month, Month, Day of the Week, and Year, with Leap-Year Compensation Valid Up to 2100
- Accuracy ±2ppm from 0°C to +40°C
- Accuracy ±3.5ppm from -40°C to +85°C
- Digital Temp Sensor Output: ±3°C Accuracy
- Register for Aging Trim
- Active-Low RST Output/Pushbutton Reset Debounce Input
- Two Time-of-Day Alarms
- Programmable Square-Wave Output Signal
- Simple Serial Interface Connects to Most Microcontrollers
- Fast (400kHz) I^2C Interface
- Battery-Backup Input for Continuous Timekeeping
- Low Power Operation Extends Battery-Backup Run Time
- 3.3V Operation
- Operating Temperature Ranges: Commercial (0°C to +70°C) and Industrial (-40°C to +85°C)

- **UBLOX Neo 7 - GPS**
  The new Ublox NEO 7 series is a high sensitivity, low power GPS module that has 56 channels and outputs precise position updates at 10Hz.

  - 56 channel Ublox NEO 7
  - GPS L1 C/A, GLONASS L1 FDMA
  - QZSS L1 C/A
  - Galileo E1B/C
  - SBAS: WAAS, EGNOS, MSAS
  - 10Hz update rate
  - 25x25x2 Ceramic patch antenna
  - Rechargeable 3V Backup battery
  - Low noise 3.3V regulator
  - I^2C EEPROM storage
  - Power and fix LED's
  - Pedestal Mount/Case
  - Pixhawk/PX4 compatible
  - LNA MAX2659ELT+



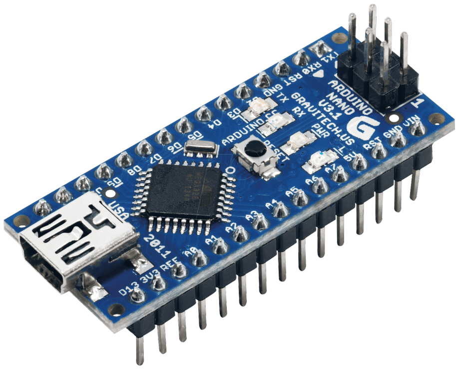
- **Dekart SIM card reader**

Dekart SIM card reader is a fully functional USB smart card reader.
Designed with modern hardware technologies, this USB SIM card reader works reliably in any operating system that includes a PC/SC driver, with any GSM SIM cards, 3G USIM cards, CDMA R-UIM cards, as well as with smart cards.

3.3.2 Arduino Nano

The very first version of the project envisaged the design of a PCB (Printed Circuit Board) with an Arduino Nano embedded with all the sensors and components described above.
The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328.
The board is designed especially for IoT applications and scenarios.

- Microcontroller: ATmega328
- Architecture: AVR
- Operating Voltage: 5 V
- Flash Memory: 32 KB of which 2 KB used by bootloader
- SRAM: 2 KB
- Clock Speed: 16 MHz
- Analog I/O Pins: 8
- EEPROM 1 KB:
- DC Current per I/O Pins: 40 mA (I/O Pins)
- Input Voltage: 7-12 V
- Digital I/O Pins: 22
- PWM Output: 6
- Power Consumption: 19 mA
- PCB Size: 18 x 45 mm
- Weight: 7 g



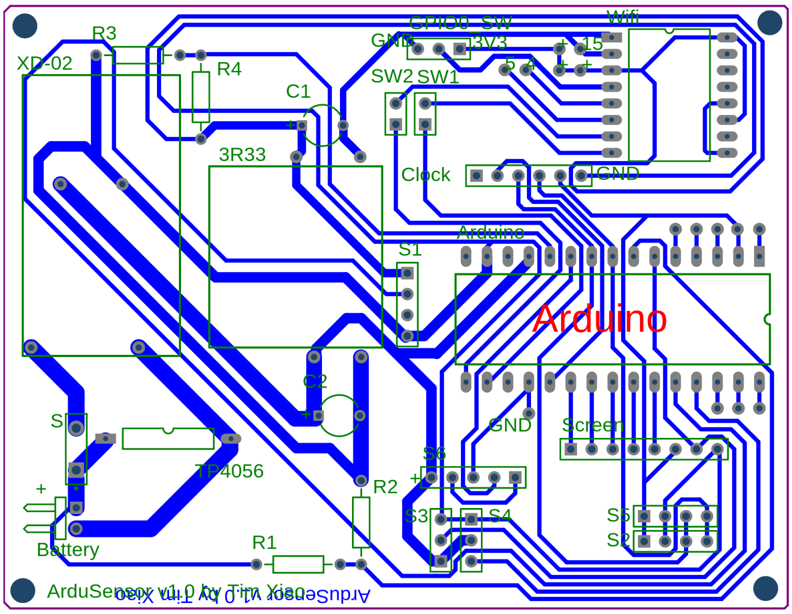


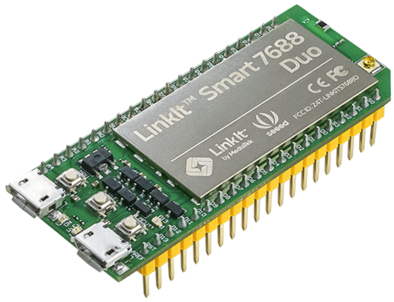
Figure 3.8: Schematic of the first version of the project, Summer 2016

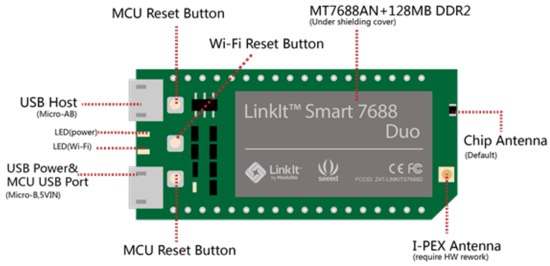
Designed in Summer 2016.

The very low-power nature of Arduino Nano lets the execution of the scripts up to 4/5 days, thanks to the sleep mode available in some Arduino boards.

This version lacks of Linux flexibility and it does not connect to WPA2 Enterprise networks or EAP-SIM. The WPA2-PSK is provided by a ESP8266 in the pcb.

The Thai sensor visible on the web page project is running this settings.





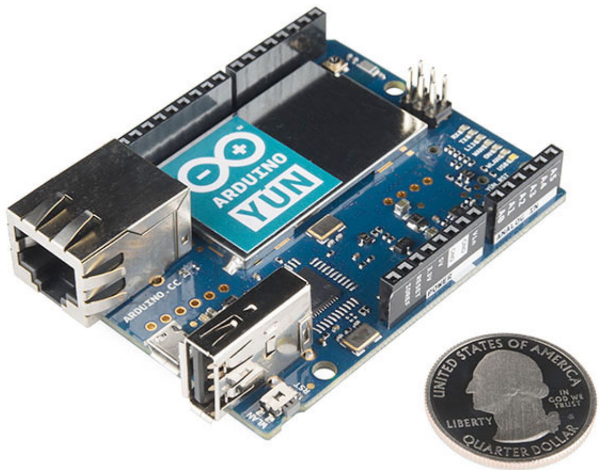
3.3.3 Linkit Smart 7688 Duo

LinkIt Smart 7688 Duo is an open development board compatible with Arduino Yún sketches, based on the OpenWrt Linux distribution, MT7688 and ATmega32u4.
The board is designed especially to enable the prototyping of IoT Application for smart house or office.
Since it is compatible with Arduino, this allowed us to use different features similar to Arduino Yún, which has helped on developing sketches.
The platform also offers options to create device applications in Python, Node.js and C programming language.

- 580 MHz MIPS CPU
- Single input single output(1T1R)
- Wi-Fi 802.11 b/g/n (2.4G)
- Pin-out for GPIO, I^2C, SPI, SPIS, UART, PWM and Ethernet Port
- 32MB Flash and 128MB DDR2 RAM
- USB host
- Micro SD slot
- Support for Arduino (ATmega32U4)

The Linkit Smart 7688 Duo has Linux embedded.
Sadly all the network is managed by some proprietary driver that don't support WPA2-PEAP Enterprise or EAP-SIM, even if this board was the most promising due to its low-power consumption.

3.3.4 Arduino Yùn



The Arduino Yún is a microcontroller board based on the ATmega32u4 and the Atheros AR9331. The Atheros processor supports a Linux distribution based on OpenWrt named Linino OS.
It consumes more than Arduino Nano but we tested it for its functionalities.

AVR Arduino microcontroller

– Microcontroller: ATmega32U4

– Operating Voltage: 5V

– Input Voltage: 5 V

– Digital I/O Pins: 20

– PWM Output: 7
– Analog I/O Pins: 12
– DC Current per I/O Pin: 40 mA on I/O Pins; 50 mA on 3,3 Pin
– Flash Memory: 32 KB (of which 4 KB used by bootloader)
– SRAM: 2.5 KB
– EEPROM: 1 KB
– Clock Speed: 16 MHz

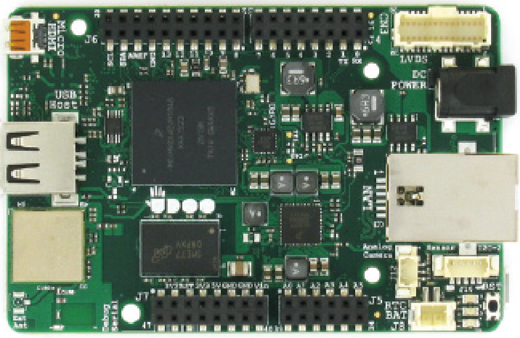
Arduino Microprocessor

– Processor: Atheros AR9331
– Architecture: MIPS
– Operating Voltage: 3.3V
– Ethernet: 802.3 10/100Mbit/s
– WiFi: 802.11b/g/n 2.4 GHz
– USB Type: 2.0 Host
– Card Reader: Micro-SD
– RAM: 64 MB DDR2
– Flash Memory: 16 MB
– SRAM: 2.5 KB
– EEPROM: 1 KB
– Clock Speed: 400 MHz

Arduino Yùn meets almost all the requirements.
The presence of a microcontroller and microprocessor allows us to have the real-time Arduino reliability together with the Linux flexibility. It is the first board tested that supports WPA2 Enterprise: we used wpa_supplicant configuration to

connect to *eduroam* network in the UPMC university.
OpenWrt lacks of our SIM reader's drivers and EAP-SIM is not supported.
It has very low flash memory for storage out-of-the-box even if there are scripts in order to allow some expansions on external SD cards.
According to our power consumption tests, Arduino Yùn consumes 460mAh ish.

3.3.5 Udoo Neo Full



Launched in 2015 and funded through Kickstarter, UDOO NEO is an Arduino-powered Android / Linux single board computer enriched with 9-axis motion sensors, Bluetooth 4.0 and a Wi-Fi module.
The unique feature of UDOO NEO is its heterogeneous processor, based on two cores embedded on the very same chip: 1GHz ARM® Cortex-A9 and an Arduino UNO-compatible platform that clocks at 200 MHz, based on a Cortex-M4 I/O real-time co-processor, all wrapped into an i.MX 6SoloX processor by NXP.

- Processor: NXP® i.MX 6SoloX applications processor with an embedded ARM Cortex-A9 core and a Cortex-M4 Core
- Memory: 1GB
- Graphics: Integrated 2d/3d graphics controller
- Video Out: Micro HDMI interface - LVDS interface + touch (I^2C signals)
- Video In: Analog camera connection supporting NTSC and PAL - 8-bit parallel camera interface

– Mass Storage: MicroSD card slot onboard - 8-bit SDIO interface

– Audio: HDMI audio transmitter - S/PIDF & I2S

– USB: 1x USB 2.0 Type A ports - 1x USB OTG (micro-AB connector)

– Networking: Fast ethernet RJ45 – 10/100Mbps - Wi-Fi 802.11 b/g/n, Direct Mode SmartConfig and Bluetooth 4.0 Low Energy

– Serial Ports: 3x UART ports - 2x CAN Bus interfaces

– Other Interfaces: 8x PWM signals - 3x I^2C interface - 1x SPI interface - 6x multiplexable signals

– Power Supply: 5 V DC Micro USB - 6-15 V DC Power Jack - RTC Battery Connector

– LEDs: Green Power Status LED - User Configurable LEDs (Red and Orange)

– Integrated Sensors: 3-Axis Accelerometer - 3-Axis Magnetometer - 3-Axis Digital Gyroscope - 1x Sensor Snap-In I^2C connector

– Dimensions: 89mm x59mm (3.50″ x 2.32″)

– Arduino Pinout: Arduino-Compatible through the standard Arduino Pins layout and compatible with Arduino shields.

– Digital I/O Pins: 32 extended GPIOs (A9 dedicated) - 22 arduino GPIOs (M4 dedicated)

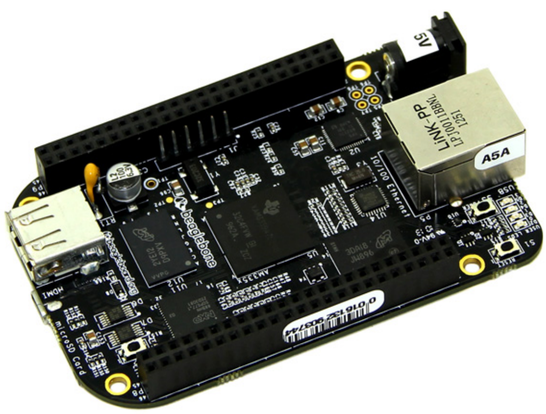
– Analog Input Pins: 6 available Pins

– Operating System: Android Lollipop & Linux UDOObuntu2 (14.04 LTS)

Udoo Neo Full is an Italian project that aims to provide the market with a low-power and low-cost single-board computer with high performances.
The presence of Linux kernel (even if based on the old version 3.14.56) together with 16GB of flash memory, allowed us to compile and install every package we needed and all the sensors drivers.
It supports WPA2 Enterprise and It is the first board tested that supports EAP-SIM. We tested the FreeWifi connection availability with the Dekart reader and a

sim card.
The only negative note concerns the power consumption: the sleep mode is not officially supported and we didn't have support or hints from the developers about it; this makes Udoo Neo consumes high in some circumstances where there are Wifi, GPS and PM sensors actived.
According to our power consumption tests, Udoo Neo Full consumes 550mAh ish.
Udoo Neo is therefore the candidate for the Thai scenario, where the nodes are fixed and we have A/C power supplier or solar panels in the buildings, without weight problems.

3.3.6 Beaglebone Black Wireless



BeagleBone Black Wireless is the newest board in the BeagleBone family that replaces the 10/100 Ethernet port with on-board 802.11 b/g/n 2.4GHz Wi-Fi.
The BeagleBone Black Wireless has a Texas Instruments WL1835MODGBMOCT module and is an open source board utilizing Cadsoft EAGLE.

The new BeagleBone Wireless introduces the BeagleBoard compatible Octavo OSD3358 system-in-package (SiP). The SiP has a Texas Instruments TPS65217C PMIC, Sitara AM335x MPU, and TL5209 LDO.

– Processor: Sitara AM335x MPU 1GHz ARM® Cortex-A8 - 2x PRU 32-bit microcontrollers

- Memory: 512MB DDR3 800MHZ RAM
- Graphics: 3D graphics accelerator
- Video Out: HDMI
- Mass Storage: 4GB 8-bit eMMC on-board flash storage
- USB: USB client for power & communications
- Networking: 802.11b/g/n and Bluetooth 4.1 plus BLE
- Power Supply: 5 V DC Micro USB - 6-15 V DC Power Jack - RTC Battery Connector
- Dimensions: 86mm (L) x 53mm (W)
- Analog Pins: 7 pins (AIN pins are rated 0 - 1.8V only)
- Digital I/O Pins: 65 pins (rated 0 - 3.3V only)
- Operating System: Debian, Arch, Angstrom

Beaglebone Black Wireless is a good candidate for the Parisian scenario.
The possibility to set its AM335X into the sleep-mode allows to reduce the power consumption drastically.
Our preliminary test estimated that the consumption is 200mAh, less than half the Udoo's consumptions.
Debian 8.6 and Linux 4.4 supply the maximum flexibility with the most recent drivers and packages support. In addition, the Kernel 4.4 has massively improved its power management.

Chapter 4

Implementation

The process flow is divided into 4 steps:

1. An Arduino sketch deals with initialize the sensors, sample the data produced and it writes them to the serial port.

2. The script `Ardu2Linux` is responsible for parsing the serial port from the Linux side, identifying which type of data have been produced; it checks if they're syntactically correct and finally it creates a new file in the file system like this:

   $$< \textbf{latitude} * \textbf{longitudine} * \textbf{altitude} * \textbf{value} >$$

   while its filename is the epoch[1] and its extension is the kind of sensor[2].
   E.g. a temperature sample produced will be name like this:

   `1482237603.temp`: 48.8628005981*2.32920002937*20*23.5

3. The script `Linux2Server` is responsible for checking if there are samples in the sampling folder defined in the settings file. If true, the script creates the TLV (Type Length Value) packets to send to a centralized server.

4. Eventually, on server-side, a script reads the incoming packets, it checks if their format is correct and it add them into the database.

[1]seconds that have elapsed since 1/1/1970

[2]e.g. temp or humi or PM_{10}

4.1 Arduino

In details, let's examine the sketch responsible for reading the sensors.
The sketch follows the Arduino template: there is `setup()` where the sensors are initialized and `loop()` where the sensors are read.

Algorithm 1 Setup sequence

```
procedure SETUP
    Set RTC Alarm
    Set the working state of the PM sensor
    Serial Arduino/Linux init
    Serial PM sensor init
    RTC init
    DHT22 init
    BMP init
```

Once setup the serial port and sensors' GPIOs, it is loop time:
at the beginning the Real Time Clock (RTC) synchronizes with the system clock, checking which one is correct.
Subsequently, the sampling gets started through some Arduino libraries; then the sketch writes on the serial port every data it collected checking if they're not `NaN`[3].
The flow provides that the sample is done every `period` minutes[4]; eventually the loop verifies if it is necessary to set the sleep-mode: if true that means that there is still more than 15 seconds to the next RTC alarm.
In the `sleep()` function, the program checks if the RTC has triggered the alarm interrupt; only it that case, the loop restarts.

[3] Not a number
[4] Usually every 5 or 10 minutes

Algorithm 2 Loop sequence

procedure LOOP

  Time Update From Serial

  $t \leftarrow$ now()

  while $now() - t < 30$ **do**

    PM sensor warmup

  while $now() - t < 60$ **do**

    Read PM1.0 PM2.5 PM10 and Formaldehyde

  Read PM average values

  Read Relative Humidity value

  Read Temperature value

  Read Pressure value

  Print PM1.0 to Serial

  Print PM2.5 to Serial

  Print PM10 to Serial

  Print Formaldehyde to Serial

  Print Temperature to Serial

  Print Relative Humidity to Serial

  Print Pressure to Serial

  $minNow \leftarrow rtc.now().minute()$

  $n\_periods \leftarrow 60/period$

  $nextAlarm \leftarrow (((minNow/period) + 1)\%n\_periods) * period$

  $diff \leftarrow ((((nextAlarm - minNow)\%60) + 60)\%60)$

  if $diff > 1||(diff == 1 \;\;\&\;\; rtc.now().second() < LIMIT\_SECONDS)$

then return $Sleep()$

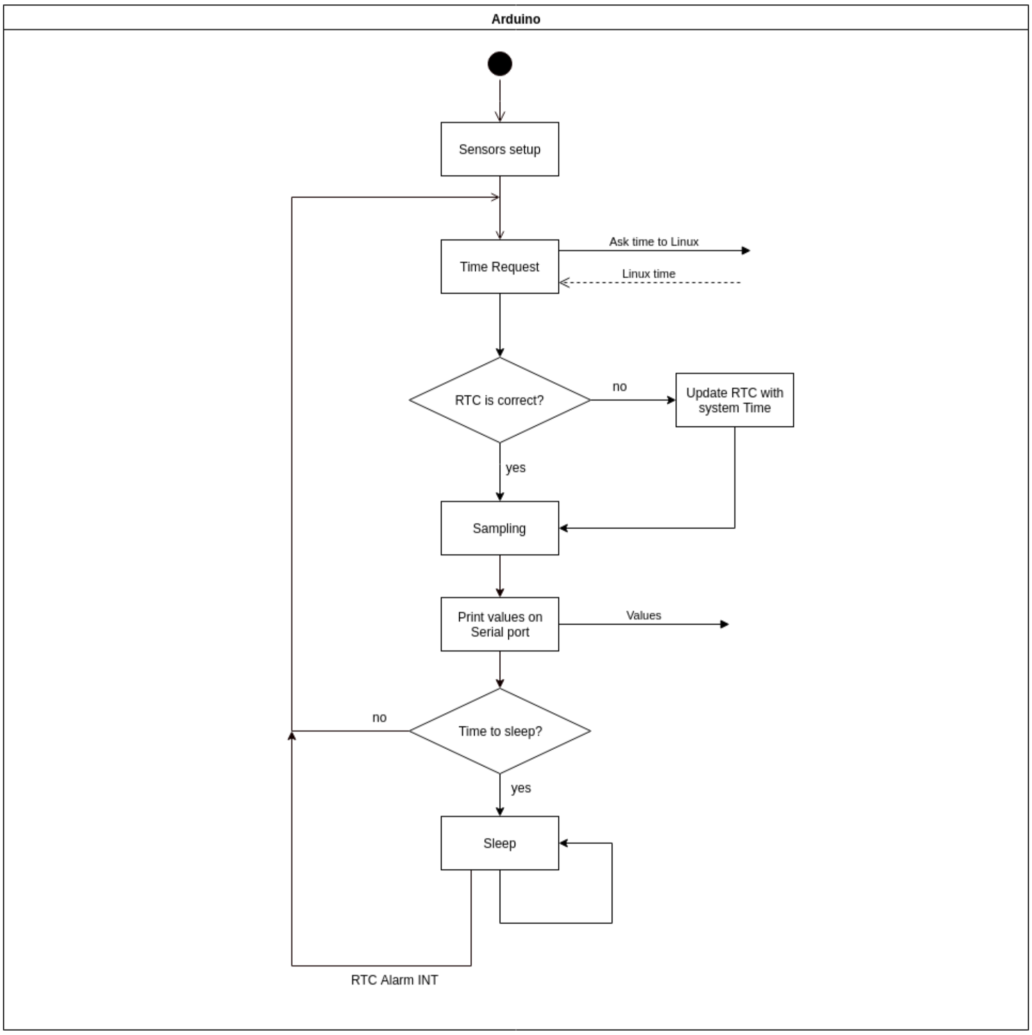


Figure 4.1: Arduino Flow chart

4.2 Linux

At Linux-side two Python scripts are responsible for parsing the serial and for building the packets ready to be sent to the server.
`Ardu2Linux` executes a series of tasks:

- Initializing the GPS serial;
- Synchronizing the system time with the RTC;

- Reading everything is written through the Arduino Serial, checking if they're syntactically correct [5].
- Depending on the data received, the script checks every value is well formed; the serial may put some noisy characters sometimes.
- At the end, a file is created in the `spool` directory defined in the configuration file.

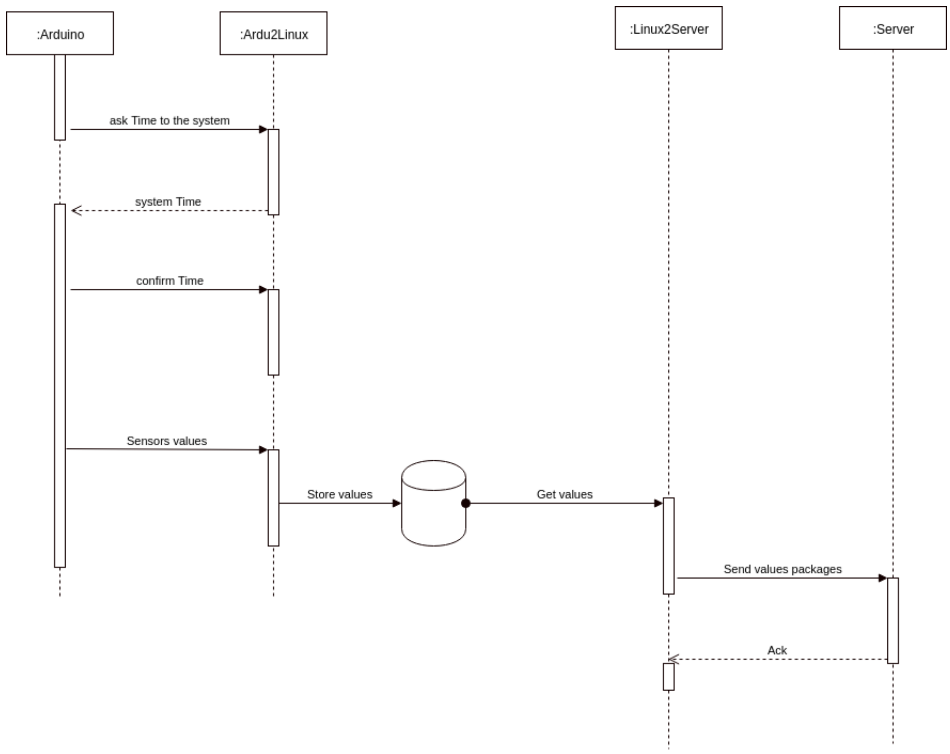


Figure 4.2: Arduino and Linux Interaction chart

`Linux2Server`, however, runs at regular intervals by Linux `CRON` and it checks if there are files available.
If it is so, it opens a socket with the server and it starts creating the TLV packets: every packet consists of an header, with the board ID and the data timestamp, and a data field, with the type of the data and the numeric/string value.

[5]Every values follows the pattern type#value

The following provides an example for temperature packet:

– **Header**: 42,1482237603

– **Data**: h,4,23.5

In this example, 'h' stands for short integer, while 4 is the temperature type id. The type IDs are defined in a specific database table described below.

Eventually, on server side, a python script is responsible for demultiplexing every incoming packet. It checks if they're well formatted and it generates the correct SQL entries to the database.

4.3 Database

The database is made up of 4 tables:

- **unified-data**: here are the sensors' values entries.
  Every entry contains the following fields:
  - `id`, of the entry,
  - `node_id`, specified in unified-node table,
  - `server_timestamp`, that corresponds to the timestamp when the server has received the sample,
  - `timestamp`, that corresponds to the timestamp when the data has been produced by the sensor,
  - `type_id`, it is the type identification specified in unified-type,
  - `value_num`, if the value is numeric,
  - `value_str`, if the value is a string;

- **unified-node**: here are the entries with every boards' id number that are unique for the nodes in the world map.
  Every entry contains the following fields:
  - `id`, of the node,
  - `name`, user-friendly node's name,
  - `interval`, (in minutes) between two samples.
- **unified-type**: here are the entry with every kind of type of data: every entry stands for a different kind of sensor.
  Every entry contains the following fields:
  - `id`, of the node,
  - `view_id`, id used to show the node on the world map,
  - `notes`, type name,
  - `datatype`, data format,
  - `multiplier`, of the value
- **unified-view**: this table contains the information useful to the front-end in order to show correctly the charts on the map.
  Every entry contains the following fields:
  - `id`, of the kind of value,
  - `name`, full name of the kind of type,
  - `unit`, unit of measurement,
  - `minimum`, minimum value in order to show the node as green,
  - `threshold`, minimum value in order to show the node as orange,
  - `maximum`, minimum value in order to show the node as red,
  - `number`, has value 1 if it is numeric, 0 instead

4.4 Front-end

We developed a user-friendly web platform to let users to watch nodes data. Through *Google Maps* is possible to identify every node connected in the world and it is possible to check the data flows in real-time.
It is not only possible access the fresh data, but changing the date, is possible to read every sensors history.

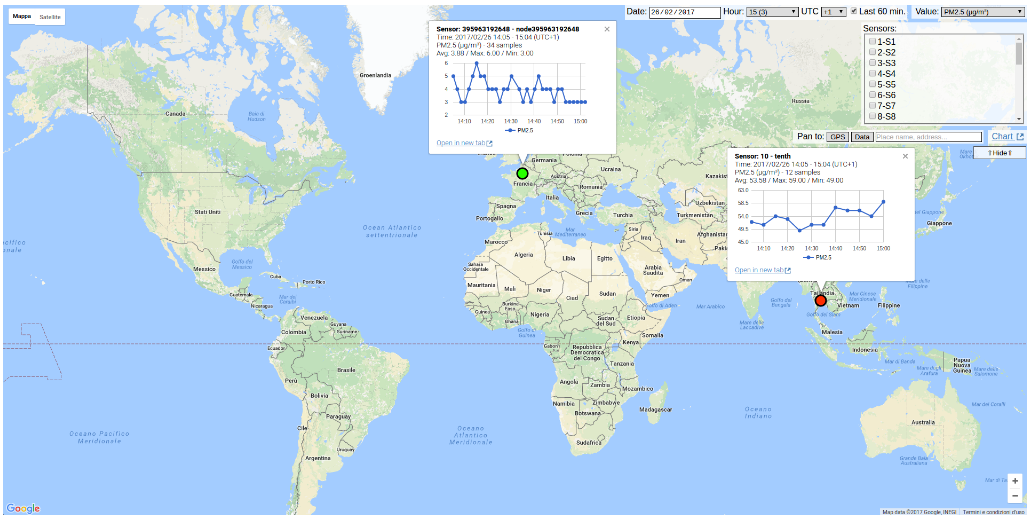


Figure 4.3: Two sensors on our Web application

Chapter 5

NDN into the project

In view of a future deployment, in order to study potential and characteristics of this new protocol, I simulated a possible NDN application use scenario through NS3 (*ns-3 Network Simulator*). The application chosen for the test is *ChronoSync*. The in-depth analysis of ChronoSync is well described in [46]

5.1 ChronoSync

ChronoSync originates from the **synchronization** chat protocols or from storage cloud service like Dropbox: it describes a dataset state through a crypted digest (e.g. SHA256); every nodes' goal is to **exchange this digest** in order to check if the state matches with its internal state or any changes happened.
Every node, through periodic *broadcast Interests*, can announce a data production or it can get the data from another producer in the graph.
Each application, based on ChronoSync, has to define a *Logic* module, where there are all the methods needed to the application logic management. (e.g. message sorting in a chat, *ChronoChat*, or files synchronization, in Cloud storage, *ChronoShare*).

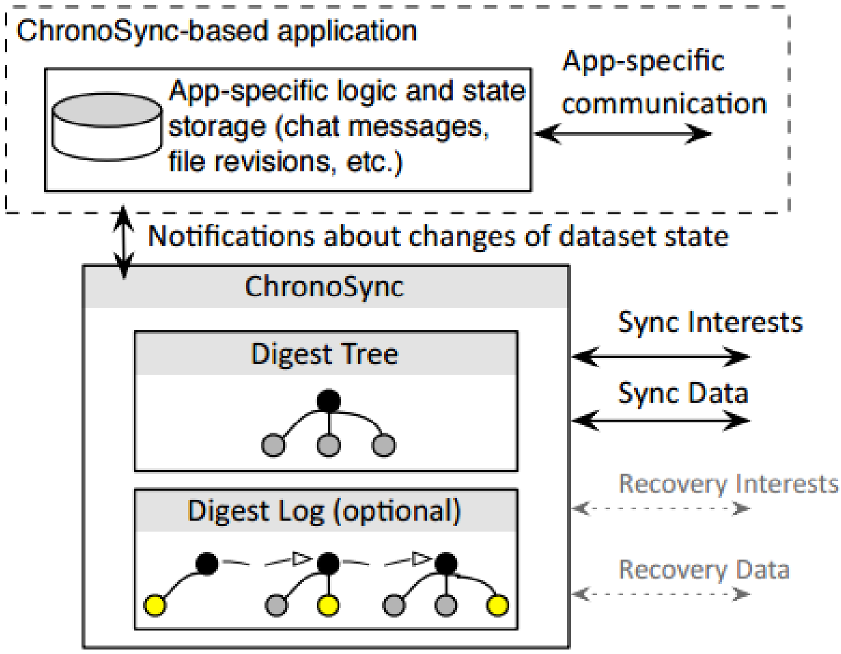


Figure 5.1: ChronoSync overview [46]

5.2 The ndnSIM simulation

With the help of NS3 and *ndnSIM* module, I defined a sensors tree, where each nodes can have two different purposes:

- **Producer node:** The leaf nodes of the tree simulate the real sensors. Every minutes they **produce a data** (in the simulation they generate a pseudo-random integer) like an environment sample in reality. Next, ChronoSync is responsible for retrieving the date from the Interest in order to aggregate it with the others. Everytime a date has been produced, a broadcast Interest is generated to announce the other nodes that the state digest has changed; so the synchronization begins.

- **Aggregator node:** The nodes inside the tree (root included) are aggregator nodes, i.e. during the simulation, they try to **get every data** has produced

from the producers, through the sync mechanism by ChronoSync. Then they start getting data processed like the average or min/max of last hour sampling.
Once the "aggregator data" has been processed, every node register the new data's name into the FIB in order to retrieve them from any nodes in the network, through an Interest package.

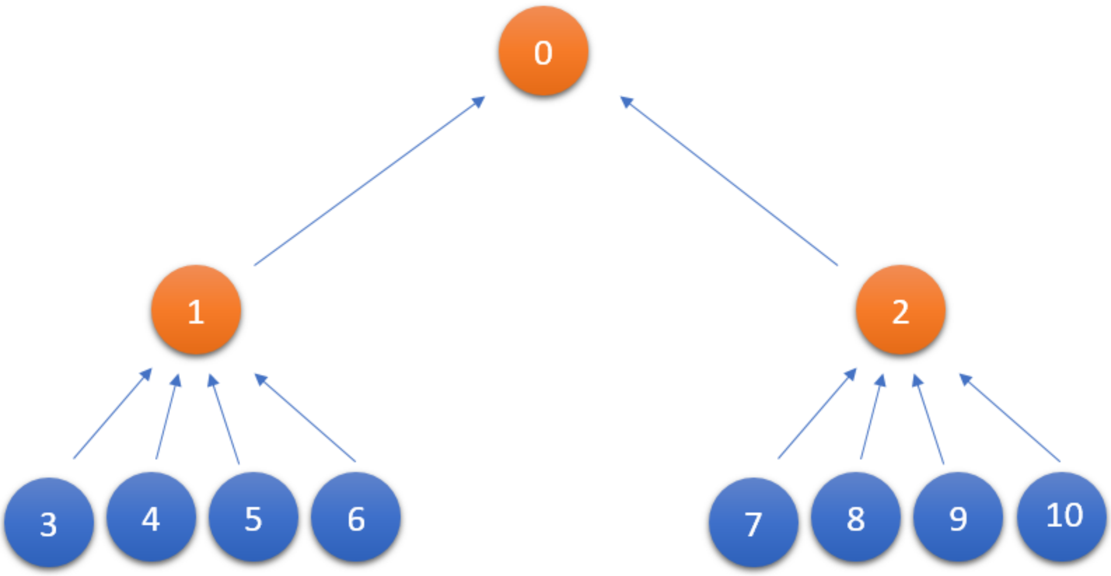


Figure 5.2: Simulation tree [46]

The simulation aims is to check the ChronoSync architecture and reliability in a semi-real context. The topology described may well represent the Thai scenario: lots of sensors sample the pollution and they write values in synchronization; at this point a coordinator node might want to check the PM average in the area, doing '*sync*' between its subtree nodes.
The higher the tree level, the more processed data are available and they will be related to a bigger area.

The simulation has the following characteristics:

- Tree topology,
- 3 Aggregator nodes, one of them directly linked to 4, 20, 50, 100 producer nodes, each,
- Producer nodes record a new value every 60 seconds,
- Every node sends a Sync Interest every 60 seconds,
- The simulation time is 1000 seconds.

5.3 Results

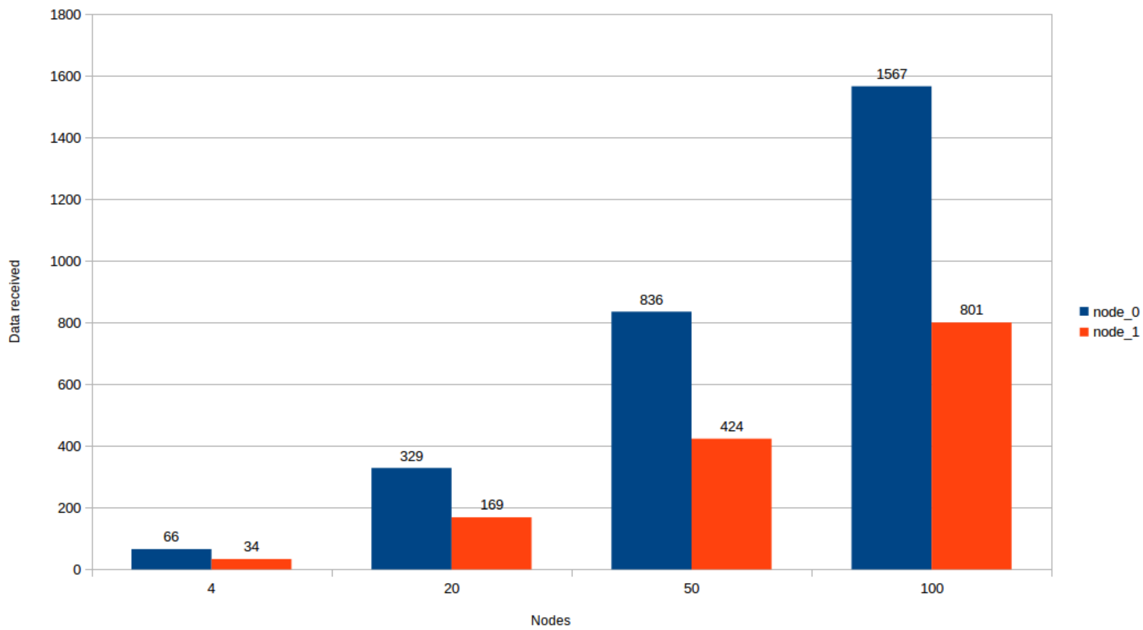


Figure 5.3: Received Packet Overtime

The first chart is the result of 4 simulations with different number of producers. It is important to that the root, *node_0*, gets all the data sent by producer nodes

properly. The root node is therefore in a position to aggregate all the tree.
It should be noted that a node 'father' (*node_1*), gets half of root's data ish, as expected since it only gets its subtree values.

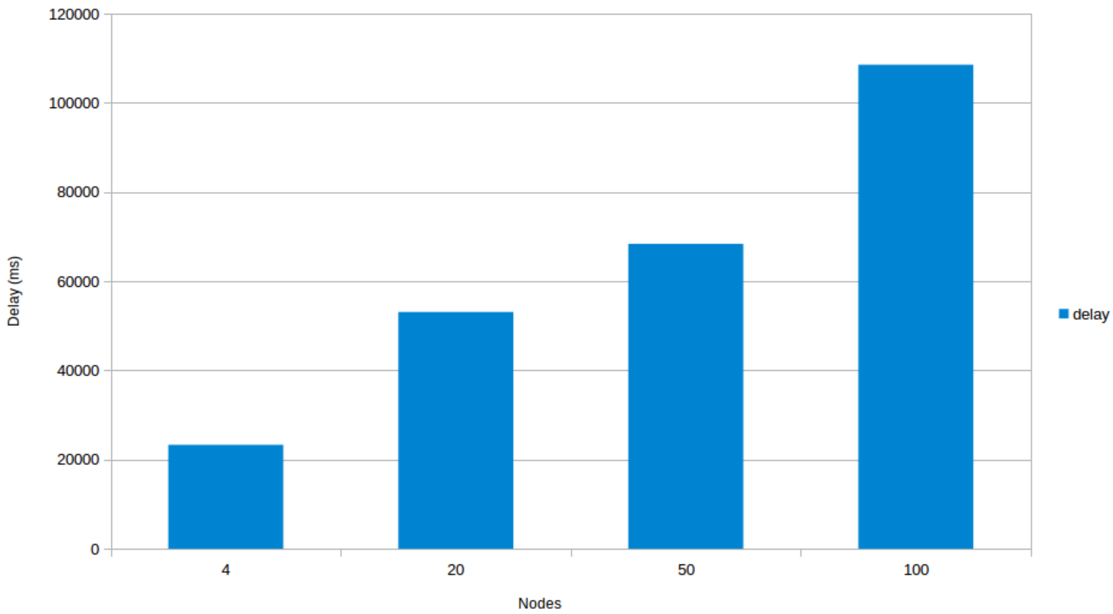


Figure 5.4: Data propagation delay

The second chart, unfortunately, shows a problem in the ChronoSync application used. In these 4 tests, the delay between a data production and its arrival to the root has been calculated.
While the delay is acceptable with values < 20, since the synchronization happens every minute, it gets worse if number of leaves increase: the delay does not remain constant.
This is due to the broadcast nature of ChronoSync, that fits better with experiments in which every node can synchronize with the others, e.g. a chat application. The aggregator nodes, in this case, end up becoming bottle-necks, slowing the Sync Interest forwarding.

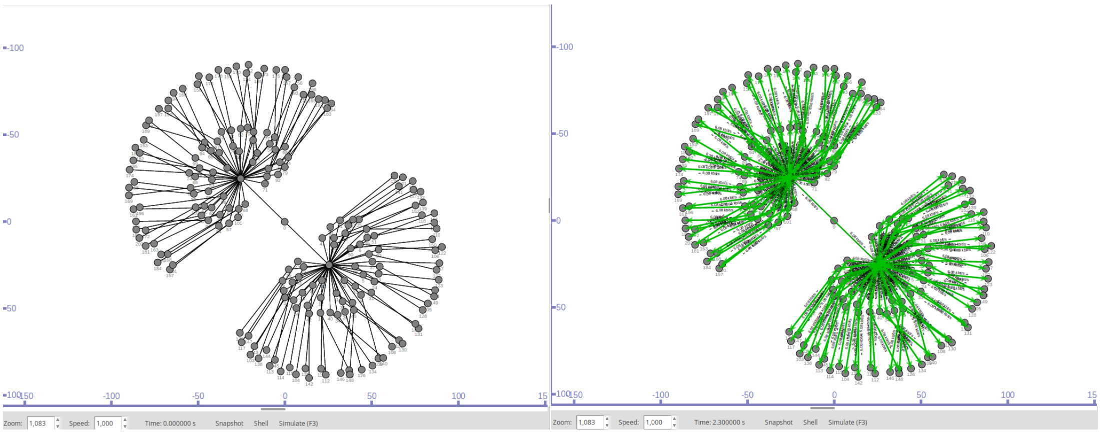


Figure 5.5: ndnSIM simulation

A further test with forwarders has been tested with the same results: moving the forward from a ChronoSync node to a NDN node, the aggregator node still gets lots of Interest packages.

Conclusions

Nowadays, the low cost of components and a wide choice of boards and operating systems, let the IoT wide open for the people's fantasy. Anyone with the right idea and with a small budget can build equipment able to solve unimaginable problems.
This thesis showed two important aspects:

- The use of particulate matter sensors with microcontrolled boards allows anyone to build a domestic single-board, able to monitor the atmospheric pollution. Most ambitious projects, like the one described in this thesis, concomitantly with specialized agencies, can help the research to make further progress even in situations unlinked to computer science, such as medical, health, and education services.

  Monitoring fire stations in remote areas in Thailand and wearable sensors among the *Paris rues*, are two example, soon ready, of how information technology can have a significant impact on human life.

- From an experimental point of view, it has not just dealt with a new concept of sensors sampling, but a new protocol has been tested instead of using traditional one, i.e. IP protocol. The use of NDN, even if it was a preliminary test and it needs more studies and attention, showed its potential allowing to access data in a new way.

Certainly, further improvements must be made in NDN scope to check its correctness and the ChronoSync reliability in a similar context, such as the one proposed by this thesis.

Bibliography

[1] Dave Evans - The Internet of Things - How the Next Evolution of the Internet Is Changing Everything, 2011

[2] K. Ashton, That "Internet of Things" thing, RFiD Journal (2009)

[3] H. Sundmaeker, P. Guillemin, P. Friess, S. Woelfflé, Vision and challenges for realising the Internet of Things, Cluster of European Research Projects on the Internet of Things—CERP IoT, 2010.

[4] J. Buckley, ed., The Internet of Things: From RFID to the Next-Generation Pervasive Networked Systems, Auerbach Publications, New York, 2006

[5] IEEE - Special Report: The Internet of Things, 2014

[6] "From the Internet of Computers to the Internet of Things" (Mattern et al., 2010)

[7] L. Atzori, A. Iera, G. Morabito, The Internet of Things: a survey, Computer Networks 54 (2010) 2787–2805.

[8] Internet of Things (IoT): A vision, architectural elements, and future directions - Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami

[9] IEEE - Towards a definition of the Internet of Things (IoT), 2015

[10] IoT Reference Model -Martin Bauer , Nicola Bui, Jourik De Loof, Carsten Magerkurth, Andreas Nettsträter, Julinda Stefa, Joachim W. Walewski, 2013

[11] Emmanuel Baccelli, Oliver Hahm. RIOT: One OS to Rule Them All in the IoT. [Research Report] RR-8176, 2012. Volume 2009 (2009), Article ID 730894

[12] An Experimental Comparison of Event Driven and Multi-Threaded Sensor Node Operating Systems, Cormac Duffy, John Herbert, Cormac Sreenan, Utz Roedig

[13] The Internet of Things: Breakthroughs in Research and Practice, Management Association, Information

[14] S. Haykin, Cognitive radio: brain-empowered wireless communications, IEEE J. Sel. Areas Commun. 23 (2005) 201–220

[15] I. Chlamtac, M. Conti, J.J.-N. Liu, Mobile ad hoc networking: imperatives and challenges, Ad Hoc Netw. 1 (1) (2003) 13–64

[16] Harbor: Software-based Memory Protection For Sensor Nodes - Ram Kumar, Eddie Kohler, 2007

[17] A Survey of the Internet of Things, Sumita Dey, Dr. S. D.Joshi, Shruti Patil, Dr. Barun Kumar, Dr. Vishwal Ajit Kagi

[18] N. Kushalnagar, G. Montenegro and C. Schumacher, IPv6 Over Low- Power Wireless Personal

[19] Cyber Security for Intelligent World with Internet of Things and Machine to Machine Communication, Vandana Rohokale and Ramjee Prasad

[20] Internet of things: Vision, applications and research challenges - Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, Imrich Chlamtac

[21] H. Karl and A. Willig, Protocols and Architectures for Wireless Sensor Networks, John Wiley and Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, England, 2005.

[22] Wireless Sensor Network Management and Functionality: An Overview, Dimitrios GEORGOULAS, Keith BLOW

[23] A New Security Scheme for Wireless Sensor Networks, Junqi Zhang and Vijay Varadharajan

[24] Wireless Sensor Networks - F.Lewis

[25] Wireless Sensor Network: A Promising Approach for Distributed Sensing Tasks - Madhav Bokare, Anagha Ralegaonkar

[26] A Survey on Applications of Smart Sensor Networks in Building a Futuristic Smarter World, Pratiksha R. Shetgaonkar

[27] A Service-Oriented Self-Configurable Middleware for Autonomous Sensor Networks, Ina Podolski, Achim Rettberg, 2009

[28] Wireless Sensor Network - Madhav Bokare, Anagha Ralegaonkar

[29] A Survey of Wireless Sensor Network Based Air Pollution Monitoring Systems, Wei Ying Yi, Kin Ming Lo, Terrence Mak, Kwong Sak Leung, Yee Leung and Mei Ling Meng, 2015

[30] Choi, S.; Kim, N.; Cha, H.; Ha, R. Micro Sensor Node for Air Pollutant Monitoring: Hardware and Software Issues. Sensors 2009, 9, 7970–7987

[31] Dobre, A.; Arnold, S.J.; Smalley, R.J.; Boddy, J.W.D.; Barlow, J.F.; Tomlin, A.S.; Belcher, S.E. Flow field measurements in the proximity of an urban intersection in London, UK. Atmos. Environ. 2005, 39, 4647–4657.

[32] FreeWifi - https://wifi.free.fr/

[33] United States Environmental Protection Agency. Buildings and their Impact on the Environment: A Statistical Summary, 2009. Available online: http://www.epa.gov/greenbuilding/pubs/gbstats.pdf (accessed on 16 June 2015).

[34] Named Data Networking - Lixia Zhang, kc claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang

[35] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. In CoNEXT, 2009.

[36] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang. A case for stateful forwarding plane. Computer Communications: ICN Special Issue, 36(7):779–791, April 2013.

[37] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang. Interest flooding attack and countermeasures in Named Data Networking. In Proc. of IFIP Networking, May 2013.

[38] Preterm birth associated with maternal fine particulate matter exposure: A global, regional and national assessment, Christopher S. Malleya, Johan C.I. Kuylenstiernaa, Harry W. Vallacka, Daven K. Henzeb, Hannah Blencowec, Mike R. Ashmorea, 2017

[39] Air Quality Guidelines for Europe - World Health Organization - Regional Office for Europe Copenhagen WHO Regional Publications, European Series, No. 91 Second Edition

[40] Burden of Disease from Rising Coal-Fired Power Plant Emissions in Southeast Asia - Shannon N. Koplitz, Daniel J. Jacob, Melissa P. Sulprizio, Lauri Myllyvirta, and Colleen Reid, 2017

[41] #MWC2017: IoT Adoption Continues to Present Security Challenges - https://www.infosecurity-magazine.com/news/iot-adoption-continues-to-present/

[42] OECD based on Verdone et al., 2008

[43] Wikimedia Commons - Particulate danger

[44] World Air Quality Index - http://aqicn.org/map/world/

[45] CNN - Southeast Asia air pollution deaths could triple http://edition.cnn.com/2017/01/12/asia/southeast-asia-pollution-coal-report/

[46] Let's ChronoSync: Decentralized Dataset State Synchronization in Named Data Networking, Zhenkai Zhu and Alexander Afanasyev

[47] Kadri, A.; Yaacoub, E.; Mushtaha, M.; Abu-Dayya, A. Wireless sensor network for real-time air pollution monitoring. In Proceedings of the 2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA), Sharjah, UAE, 12–14 February 2013; pp. 1–5.

[48] Mendez, D.; Perez, A.J.; Labrador, M.A.; Marron, J.J. P-Sense: A participatory sensing system for air pollution monitoring and control. In Procedings of the 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), Seattle, WA, USA, 21–25 March 2011; pp. 344–347.

[49] By Yanjinsuren24 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=47648778