Exploring Computing Continuum in IoT Systems : sensing, communicating and processing at the Network Edge

Davide Aguiari

Università degli
Studi di Cagliari

Ph.D. in Electronic Engineering and Computer Science
Cycle XXXIII

ED 130 - Informatique, télécommunications et électronique de Paris (EDITE)

Ph.D. Thesis

Exploring Computing Continuum in IoT Systems: Sensing, Communicating and Processing at the Network Edge

S.S.D. ING-INF/03

PhD Student

Davide Aguiari

Supervisor

Prof. Luigi Atzori

CO-Supervisor

Prof. Giovanni Pau

Final examination academic year 2019/2020
Thesis defence: March 2021 Session

Abstract

As Internet of Things (IoT), originally comprising of only a few simple sensing devices, reaches 34 billion units by the end of 2020, they cannot be defined as merely monitoring sensors anymore.
IoT capabilities have been improved in recent years as relatively large internal computation and storage capacity are becoming a commodity.
In the early days of IoT, processing and storage were typically performed in cloud. New IoT architectures are able to perform complex tasks directly on-device, thus enabling the concept of an extended computational continuum.
Real-time critical scenarios e.g. autonomous vehicles sensing, area surveying or disaster rescue and recovery require all the actors involved to be coordinated and collaborate without human interaction to a common goal, sharing data and resources, even in intermittent networks covered areas.
This poses new problems in distributed systems, resource management, device orchestration, as well as data processing.
This work proposes a new orchestration and communication framework, namely C-Continuum, designed to manage resources in heterogeneous IoT architectures across multiple application scenarios. This work focuses on two key sustainability macro-scenarios: (a) environmental sensing and awareness, and (b) electric mobility support. In the first case a mechanism to measure air quality over a long period of time for different applications at global scale (3 continents 4 countries) is introduced. The system has been developed in-house from the sensor design to the mist-computing operations performed by the nodes. In the second scenario, a technique to transmit large amounts of fine-time granularity battery data from a moving vehicle to a control center is proposed jointly with the ability of allocating tasks on demand within the computing continuum.

Abstract

L'Internet des objets (IoT), ne comprenant à l'origine que quelques dispositifs de détection simple, atteint aujourd'hui 34 milliards d'objets connectés d'ici fin 2020. Ces objets ne peuvent plus être définis comme de simples capteurs de surveillance. Les capacités de l'IoT ont été améliorées ces dernières années tandis-que que les capacités de calcul et de stockage de masse sont devenus des marchandises.
Aux débuts de l'IoT, le traitement et le stockage étaient généralement effectués dans le cloud. Les nouvelles architectures IoT sont capables d'exécuter des tâches complexes directement sur l'appareil, permettant ainsi le concept d'un continuum de calcul étendu.
Les scénarios critiques et temps réel, comme par exemple la détection de véhicules autonomes, la surveillance de zone ou le sauvetage en cas de catastrophe, nécessitent que l'ensemble des acteurs impliqués soient coordonnés et collaborent sans interaction humaine vers un objectif commun, partageant des données et des ressources, même dans les zones couvertes par des réseaux intermittents. Cela pose de nouveaux problèmes dans les systèmes distribués, la gestion des ressources, l'orchestration des appareils et le traitement des données.
Ce travail propose un nouveau cadre de communication et d'orchestration, à savoir le C-Continuum, conçu dans des architectures IoT hétérogènes à travers plusieurs scénarios d'application. Ce travail se concentre pour gérer les ressources sur deux macro-scénarios clés de durabilité : (a) la détection et la sensibilisation à l'environnement, et (b) le soutien à la mobilité électrique. Dans le premier cas, un mécanisme de mesure de la qualité de l'air sur une longue période avec différentes applications à l'échelle mondiale (3 continents et 4 pays) est introduit. Le système a été développé en interne depuis la conception du capteur jusqu'aux opérations de mist-computing effectuées par les nœuds. Dans le deuxième scénario une technique pour transmettre de grandes quantités de données, entre un véhicule en mouvement et un centre de contrôle est proposé. Ces données sont de haute granularité temporelle relatives et permettent conjointement d'allouer des tâches sur demande dans le continuum de calcul.

Contents

Chapter 1

Introduction

Many cloud service providers, such as Amazon Web Services (AWS), Google Cloud Platform and Microsoft Azure, offer various solutions for processing and storing data in huge data-centers spread all over the world. The steadily increasing human involvement in technology, along with automated Internet of Things (IoT) sensors and actuators, require those data-centers to make an enormous effort in terms of resource allocation and availability, infrastructure management, and data provisioning.

Many new challenges have emerged in the computing scenario during the last decade: autonomous vehicular (AV) applications have changed dramatically due to the progress of artificial intelligence (AI) models embedded in the cars; Industry 4.0 is making smarter next-generation industrial robots that require enormous computing power and capacity; 5G is progressively impacting manufacturing[1], mobile, eHealth (i.e. digital medicine), smart cities, smart homes, and, overall, improving the quality of life around the world. Lower latency and high-speed bit-rate are paving the way for new and exciting applications yet to be discovered.

Such progress comes with more and more data generated by billions of sensors active at the moment; for example, in the autonomous driving arena, Intel estimates that each driver-less vehicle will produce over 4 TeraBytes (TB) of data each day[2]. While most of these data is consumed in-car, cooperative autonomous vehicles will have to exchange some percentage of these 4TB and eventually offload some computation/data to the local edge or the core cloud. Nevertheless, data are generated at a greater pace than can be consumed by humans, and IoT devices storage and, oftentimes, computing capabilities are underused.

Classic cloud computing was design to manage a certain number of homogeneous cluster nodes from a single location, whose data stored in well-established security perimeters. However, at present, these tools are incapable of tackling the complexity of a high degree of heterogeneity (i.e. a massive number of managed devices) albeit still operating in some specific scenarios. [3] Indeed, today's IoT devices differ across multiple aspects: resource allocation strategies, connectivity protocols support, location interoperability, and power management. Besides, AI-centric devices often adapt themselves to the environment and operate autonomously. Heterogeneity in

the IoT panorama also translates into connectivity issues (e.g. unpredictable channel conditions, handovers, and churn) together with the limited battery capacity of mobile devices.

Are there alternatives to the classical cloud-based approach? Is it possible to exploit the wealth of spare resources rather than directly offload the whole data to the server?

Decentralized computing is not a recent topic: the origin of edge computing can be traced back to the 90s when Akamai launched the first content delivery network (CDN)[4]. More recently, CDN evolved into:

- Pervasive Computing, in 1997, with speech recognition and cyber foraging for improved energy consumption[5];
- Peer to Peer (P2P), in 2001, for proximity-routing in overlay P2P networks to avoid long-distance hauls[6];
- Cloud Computing, in 2006, with the Amazon's Elastic Compute Cloud (EC2) service[7];
- Cloudlets, in 2009, which offered lower latency and permitted to offload work[8];
- Fog Computing, in 2012, introduced by Cisco, focused on scalability and latency in IoT[9].

Edge computing offers an answer to those applications needing predictable network reliability and low-latency data processing. The IoT nodes exploit geographically closed neighbors for tasks offloading, reducing distances and latency, and sharing tasks complexity amongst different nodes, according to the *divide-et-impera* principle. Practically, edge computing devices can ask neighbors for their capabilities and availability, as well as remote callable functions; once the approval has been received, the task supervisor asks for memory and CPU computation, thus easing the internal heavy load. Knowing *when, where, and how* an edge resource can be accessed, is often critical, and there is no prior information about units availability for executing parts of a service.

Concretely, transitioning towards edge computing does not imply abandoning the core cloud[10]. Instead, it can be used as a last resort to reduce latency and, therefore, leverage the locality of applications to increase global system capacity. Telecommunications companies (telcos) and technology providers are actively cooperating towards redesign the Cloud as an enabler for the *Network As a Platform* (NaaP), which would control the Radio Access Networks (RAN) that connects homes, businesses, and mobile devices to the Internet. The network shifts from a bare connecting medium into an operating part of the cloud. This phenomenon is also known as *Democratizing the network edge*[11]. The aim of cloud providers, in turn, is to serve the next generation of edge applications by building edge clusters

in metropolitan areas to reduce latency and increase bandwidth; on the other hand, network operators leverage cloud technology to co-locate edge applications in the access network. Edge computing also enables new mobile ways of accessing network functions, media content, and applications, thanks to frameworks like mobile edge computing (MEC)[12].

Internet service providers and hyper-scale cloud providers are jointly focusing on two application areas[13]:

(i) Near-Edge, leveraging on existing telcos' infrastructures as, for instance, Amazon AWS Wavelength;

(ii) Far-Edge, also referred to IoT Edge, makes use of MEC frameworks deployed in a location further from the cloud-data-center but closer to the end user (i.e. near the cell phone towers).

Near-Edge is usually located between the cloud data centers and the Far-Edge; it often hosts CDN caches or Fog servers to manage the CDN processes and, at the same time, temporally store sensed information.

Several architectural aspects must be considered when edge computing is adopted: (a) network connectivity, to ensure the link between the different nodes; (b) a common cooperative protocol for resources demanding; (c) containerization and virtualization, for runtime data offloading and processing; (d) secure environment, to avoid remote malicious attacks. Sharing resources and co-locating network functions and application services cannot take place without considering the risks of data privacy or integrity breaches. Delegating external nodes poses non-trivial challenges at both the digital and physical level. The objective of this thesis is, therefore, to formalize a new orchestration and communication edge computing framework, **C-Continuum** (Chapter 9), designed to overcome all the aforementioned challenges and Quality of Service (QoS) requirements. According to [10], an Edge Compute Framework shall support functions that include the following modules:

- **Protocol Adapter**, a module to standardize incoming sensor data format and outgoing actuator commands;
- **Information Broker**, an internal storage module to save samples locally
- **Rules Engine**, a module for messages coordination based on predefined rules;
- **Use Case Specific**, or rather, a dedicated module for the specific features supported by the end-user devices;
- **Management and Security**, a module that allows scalability with more and more nodes and modules.

To show its capacity and feasibility, we exploit two main macro-scenarios, thoroughly analyzed during the last three years: a global scale environmental sensing

devices network, namely **Canarin** (Chapter 3), and a **battery data management from electric vehicles** (Chapter 7).

This thesis begins outlining a *Smart Mobility* (Chapter 2) real case scenario where we take advantage of a fleet of low-cost multi-sensor platforms (Canarin) we designed from scratch in the last four years. Canarin constitutes a testbed for C-continuum. The project was born as a joint effort of Sorbonne Université, the Macao Polytechnic Institute, and the Asian Institute of Technology (Thailand). It dates back to 2016 and has been a fruitful source of numerous air pollutants data over the past few years. Originally designed as a prototype to be embedded into user accessories, or community-based stations [14], it sets the stage for the current **Canarin Nano**, which was been produced in more than 100 units. The Canarin units are currently working across three continents and seven countries. The original requirements were:

1. Collecting several pollutants and environmental factors (i.e. temperature, particle matters (PM), carbone dioxide (CO_2), etc.);
2. Flexibility, as the ability to add and remove sensors on demand;
3. Geo-located data;
4. Opportunistic data upload throughout the day;
5. Long operation life, even over multiple days;
6. Portability, in size and weight
7. High data accuracy

Once the platform got the certification from Airparif[1], the French air-quality monitoring agency for the city of Paris, we decided to enhance the whole architecture by developing our proprietary Printed Circuit Board (PCB) and an in-house 3D-printed polylactic acid (PLA) box. The new architecture supported WPA/WPA2-Enterprise networks, too. Finally, we miniaturized the Canarin to make it even more portable, to be easily attached to a backpack, and we equipped it with a 4G module. Chapter 3 and chapter 4 describe the Canarin's architecture in detail, from the sensing to the database management. The following chapters define two very different Canarin scenarios: a sensing campaign in Cesena (Italy) to monitor pollutants in the new 30.000 m^2 wide information technology (IT) university campus (Chapter 6), and the on-going measurements we are leading in the Biblioteca Joanina, a baroque library situated in the University of Coimbra (Portugal), added to the UNESCO world's heritage list in 2013 (Chapter 5). Chapter 7 illustrates the second C-Continuum application scenario: fine-time granularity battery data collection from a Nissan Leaf 2018 and the need for allocating stored samples on demand.

[1] `https://www.airparif.asso.fr/`

Unlike the environmental campaigns, here C-Continuum deals with measurements obtained by a high-speed Controller Area Network (CAN) bus placed inside electric cars. A reverse-engineering analysis was carried out through the CAN raw messages to understand how the electronic control units communicate with each other inside the vehicle. Further, we detected and collected several useful variables employing an on-board diagnostic version two (OBD2) platform 4G-connected 24/7 inside the test cars; C-continuum checks all the samples for internal aggregation, validation and collection to support the LibER[2] (*Lithium Battery for Emilia Romagna*) project at the Electric department of the University of Bologna. Chapter 8 specifies the details of one of the C-continuum's building blocks, namely X-Fi, a system that enables multi-access network for vehicles. Finally, in Chapter 9 the overall C-continuum architecture is presented along with its design implementation, analyzing pros and cons of this approach together with its compatibility with the Named Data Networking (NDN) paradigm[15].

[2] `https://liberproject.eu/`

Chapter 2

Smart Mobility and Sensing: case studies based on a Bike Information Gathering Architecture

This chapter discusses PUMA, a *Personal Urban Mobility Assistant* designed to overcome classic limited set of means of transports offered by mapping services. PUMA aims to let the user add different factors of personalization, such as sustainability, street and personal safety, wellness and health. PUMA was also the prelude to the Canarin project, described in chapter 3. In this scenario, the sensor was equipped on a smart bike, as a mean to collect data about the urban environment. The *Bike Information Gathering Architecture* (BIGA) is also described herein and it may be seen as a preliminary deploy attempt to the Canarin's Cloud Architecture, designed only at a later time.

2.1 Introduction

Smart mobility is playing a strategic role in our daily life in the urban scenario, taking into account that most people live in cities [16, 17]. Thanks to the wide diffusion of mobile devices, several services and applications based on the geographical position of users are now available [18], which support citizens while they move across the urban environment [19]. In this context, personalization can be a key factor, enabling independence of citizens, despite some specific conditions (i.e. disabilities) [20] or means of transport [21]. Existing mapping services (e.g. Google Maps) and travel planners (e.g. OpenTripPlanner, Graphhopper) provide multimodal paths computed by the same algorithms, on the basis of the same elements: time, distance, and a limited set of means of transports (e.g. cars, public buses and metros, feet). However, it is not possible to add different factors of personalization,

in terms of sustainability, street and personal safety, wellness and health, mood and satisfaction, accessibility. In this context, our idea is to design a system which acts as a Personal Urban Mobility Assistant (PUMA), supporting citizens in:

- collecting different information about the urban environment (by means of crowdsourcing and crowdsensing activities) in terms of: pollution, traffic, safety, health and fitness, etc.;
- exploiting gathered data, proving multimodal and multipreference paths in the urban environment.

In order to provide multimodal and multipreference paths in urban environments, a detailed mapping of all the elements that affect these factors (e.g., data about pollution, urban barriers and facilities, street lights, data about car accidents, data about crimes, etc) is needed. Moreover, given out this information, it is necessary a system that lets each user customize and modulate the route computation on the basis of his/her own needs, instead of using the same algorithm for all.

As regards the mapping, our approach is based on an open and participatory sensing and mapping system, with low cost sensors, which exploit users' devices too, in a common and shared data repository. We would take advantages by the potentialities of cloud architecture to create a modular open and crowdsourced system. In our work, we also tackled the following challenges:

- To introduce an innovative users' approach towards mobility choices that matches all impact factors for transportation, driving different transportation services, from single to shared, going next to the common existing booking systems, offering a social environment to share experiences and information on sustainable mobility and participate to challenges, info on traffic, lane condition and pollution [22], with the aim of supporting and improving eco-driving and sustainable behaviours [23].
- To develop a smart urban approach to mobility based upon way of booking transport systems that also take into account the carbon footprint [24].
- Integration of sustainable fleets with public transportation (i.e. buses and train) with the possibility to buy tickets by smart payment systems too.
- Data storage and data management (integration from different data sources: from public transportation and route conditions to air pollution obtained by sensors installed on bicycles or other vehicles [25], integration with traffic info).

In this chapter, we focus on a specific means of transports: bikes [26]. Bikes can be equipped with different kinds of sensors and can be connected each other, so as to create a specific vehicular network, integrated with the urban infrastructure [27, 28] and networks [29] thanks to a cloud architecture. The chapter also describes

the system architecture and details personas and related scenarios, showing how it can be exploited by different users, with different needs and preferences, applying an *altruistic IoT* approach [30].

The remainder of the chapter is structured as follows. Section II describes the system architecture. Section III defines some personas and Section IV presents some use scenarios. Finally, Section V concludes the chapter highlighting some final remarks and future work.

2.2 Cloud Architecture

In this section, we introduce our system architecture, specifically thought for bicycle vehicles, named BIGA (Bike Information Gathering Architecture), shown in Figure 2.1. Our previous work [21] focused mainly on the adoption and implementation of a specific software engineering model that envisioned every component of a mobility application as a service; the reference model was based on microservices, therefore the SMAll architecture was tailored at proposing the definition of an open and standard interface for service access. Instead, BIGA architecture describes at high level the physical and software architecture that might be put in place to provide smart bicycle services. This means that BIGA might be adopted to host and provide the implementation proposed in the SMAll project. Bikes are equipped with sensing devices capable of gathering different kinds of information (data) not only from the environment (e.g., air pollution [14]), but also from the bike itself (e.g., traveled kilometers via odometer). Once (periodically) collected, such information are sent to an entity devoted either to provide connectivity or forwarding data to the cloud via Internet; this entity might be an infrastructure located along the road, such as for example a roadside unit (RSU), or a specific gateway. Cloud infrastructure is where data are processed, stored and made available for being consumed by multiple users.

The idea is to allow users, who are interested in gathering information, to personalize a plan for a given path, depending on their daily habits or needs. The cloud hosts the software that provides path customization and other useful services, but targeted for different uses, as described in the next section. This implies that different applications might require different ways of data collection. Therefore, multiple users can source information by accessing, for example, a web application in order to properly plan in advance their path, or use a mobile application on the smartphone not only for a priori decision, but for real time consultation as well. This means that data can be consumed prior and/or during the journey. At the end of the journey, users can decide to share their experience, i.e., share "collected information" along the path; this would allow to enrich databases with new information, thus resulting in improved bikers experience that can benefit of feedbacks coming from the community. Therefore, with our approach, users are both consumer and producer, so that data are both gathered and disseminated from/in the community.

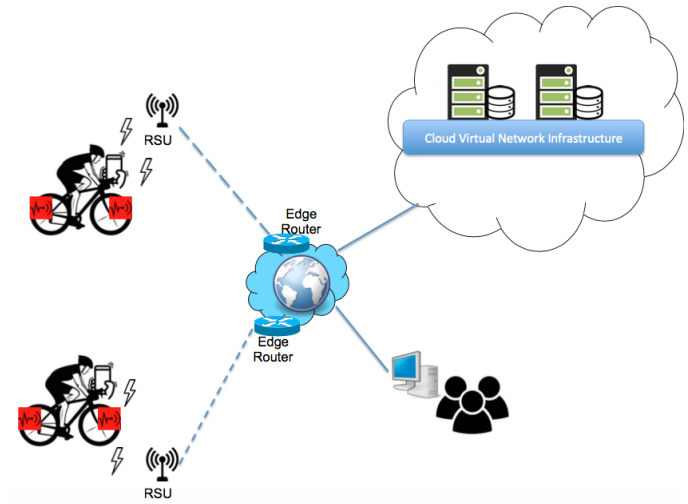

Fig. 2.1: BIGA System Architecture

The vision is that the Municipal District of Bologna (MDB) might act as service provider, i.e., providing to citizenship such "smart bicycles sharing" system. Smart bicycles would be equipped with devices targeted for the different applications (e.g., air pollution monitoring, fitness monitoring, personal safety and carbon footprint). MDB would then rent cloud resources at an infrastructure provider, whose goal would be to provide computational, network and storage resources to have service in place, besides the mobile and web application needed to interact with the service by remote users. Making the cloud hosting the applications, and making these applications available to customers, allow our "Bike as a Service" to fall under the hat of Software as a Service (SaaS). Indeed, cloud approach adoption brings several benefits:

- MDB has no need to install and run applications on their own computers, resulting in less expense in terms of buying new hardware, infrastructure provisioning and consequent maintenance.

- Other emergent paradigms might be put in place on need: Network Function Virtualization (NFV) and Software Defined Networking (SDN) might be adopted in synergy with cloud in order to provide flexible, programmable and cost effective solutions [31]. For example, software applications might run on a Virtual Machines (VMs) interconnected by a virtual network [32]; NFV would help in delivering services as virtual functions, while SDN would help in flexibly managing the (virtual) network [33].

- Cloud approach also calls for service on-demand model, that is, virtual functions could be instantiated, removed or migrated across the network without the need of deploying new hardware.

2.3 Personas

In this section, we present three personas designed with the aim of defining scenarios exploiting different characteristics of our system.

Wei

Wei is a Chinese-American visiting Scholar, working to a joint research in Bologna for three months. His research interests are related to climate change and he is very committed in reducing his individual carbon footprint. These days in Bologna are mainly devoted to work and complete all the scheduled research tasks and experiments. In his free time, he likes to go around the city centre and explore the old town of Bologna (Figure 2.2).

Wei works at the University of California, San Diego since 2007. He is in Bologna now for a joint research on climate change in the Department of Biological, Geological and Environmental Sciences of the Bologna University. Wei's family is based in S. Diego, where his wife Xiu Ying and his two children (Sean and May) live. They keep in touch with a daily call and Wei sends them a lot of pictures taken wandering around the city while he rides his bike. While in Bologna, Wei lives in the University guest quarters (Residenza di San Giovanni in Monte) located in a prominent monumental complex belonging to the University of Bologna, in the old town centre of Bologna. Wei uses a good trekking bike, loaned by a colleague from the Department for his stay in Bologna. He sporadically uses bus and other public means to reach destination that are too far from the city centre to be reached by bicycle. This responds both to Wei commitment to use sustainable transport and to his travel needs.

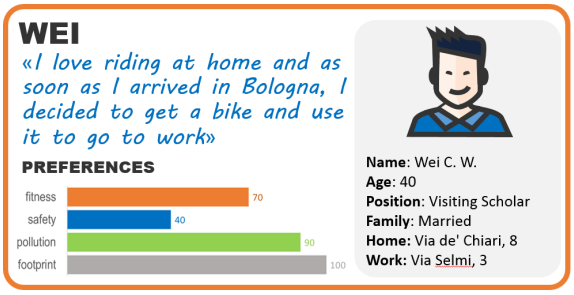

Fig. 2.2: Wei

Sven

Sven is a Swedish Erasmus student, living in Bologna for 6 months to complete his master thesis. He is vegan and he is obsessed with fitness. Stay in very good shape, eat vegan and whole food, have a generally healthy life style are very important

goals to Sven. During his stay in Bologna, Sven goes to the Department to work to his thesis, to the gym to do his workout, without forgetting to go around with friends, having fun and enjoying the city life as all students do.

He is studying Film Directing at the Faculty of Fine, Applied and Performing Arts at the University of Gothenburg and he is completing his thesis on the Kill Bill movie series by Quentin Tarantino, working in the Department of Arts of the Bologna University. Sven is single. His family of origin lives in Hästevik, a small town near Gothenburg. They keep in touch on a weekly basis with a conference call. While in Bologna, Sven lives in a shared apartment in a neighbourhood outside the city centre. He decided for this location to share the room with his friend Hugo, who is taking his master degree in Economics and Finance in Bologna. The apartment is quite near to the Business and Economics School, where Hugo studies, but pretty far (about 4 Km) from the Arts Department. Sven bought a cheap used mountain bike from another Erasmus student leaving Bologna few days after his arrival. He uses a mix of bike and bus to move around the city, depending from weather, time of the day, distance of the destination, but the bicycle is the most used mean to go to the Department and to the Gym on a daily basis because it represents an opportunity to do more workout and also to save money.

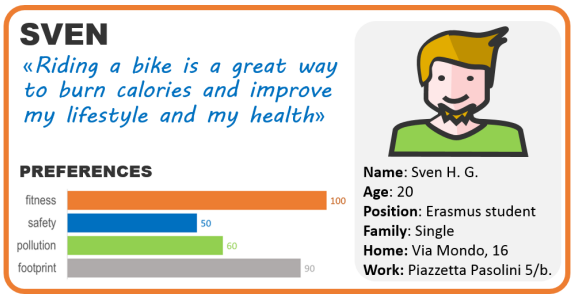

Fig. 2.3: Sven

Elena

Elena works full time at the University of Bologna. She was born in Bologna and she grow up in its city centre. Since her husband works in a nearby city, she is in charge of managing their son Tommaso (Tommy) and travel with him to and from school, or to and from her parents' house. While Elena prefers to use the car to reach points of interest outside the centre, when she goes downtown (to work, to Tommy's school and to her parents' house), she prefers to leave the car at home. Elena likes to ride the bike, but she is very worried for Tommy, by the safeness of the travel, weather issues and pollution that can be dangerous, especially during the winter. Elena works for the University of Bologna since 2008. She works in the International Desk, providing information to students wishing to enroll at the University. She works since 8.30 AM to 4.30 PM for 5 days a week, having a fast

lunch in the office nearby. Elena is married with Alberto since 2010 and they have a son, who is 4 years old. Tommy goes to a primary school (Betti Giaccaglia Plesso 2) located in the Montagnola Park, immediately near the Bologna Station. Elena got the option to enroll Tommy in this school because her father and mother live nearby and they are used to pick up Tommy from school every day at 4 PM. Tommy waits for Elena in grandparents' house, located in via Mascarella, for about an hour, to come back home with her. Elena lives in Bolognina, a neighbourhood outside the city centre, quite near to the train station. His husband, Alberto, works in Cesena and this location was chosen mainly to meet his need to easily reach the station. The place is not far from Tommy's School (about 1 Km) and from Elena workplace (about 2 km), hence Elena uses a new red city bike, fully equipped with lights and reflectors, to enhance safeness of the travel, and with a baby seat on the back.

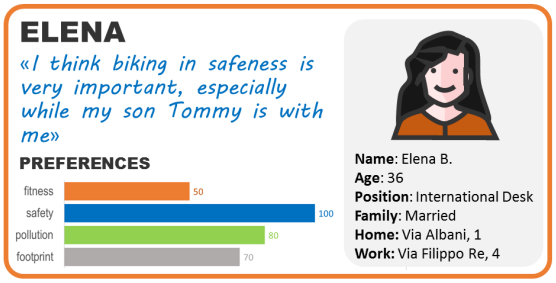

Fig. 2.4: Elena

2.4 Travel scenarios

Travel scenarios related to the personas introduced in the previous section are described in the following subsections.

Wei

Wei is going to the weekly meeting of the research team, to reschedule some late experiments. It's a foggy day, but despite cold and humid, Wei is happy to take the bike. The meeting will be at 9.00 AM, Wei is leaving the University guest quarters early, so as to have a sweet breakfast in a bar near the Department without being in a hurry. Having more time than what is strictly required to reach his destination, he decided to enjoy the ride and cross the city mainly passing through restricted traffic zones.

In Figure 2.5, there are two different paths between Wei's starting point (Via de Chiari) and destination (Via Selmi). On the left side, there is the default and shortest path proposed for bikes by GraphHopper. On the right side, there is the personalized one based on the user's preferences, computed by our PUMA. This latter avoids one of the most congested and polluted roads of Bologna.

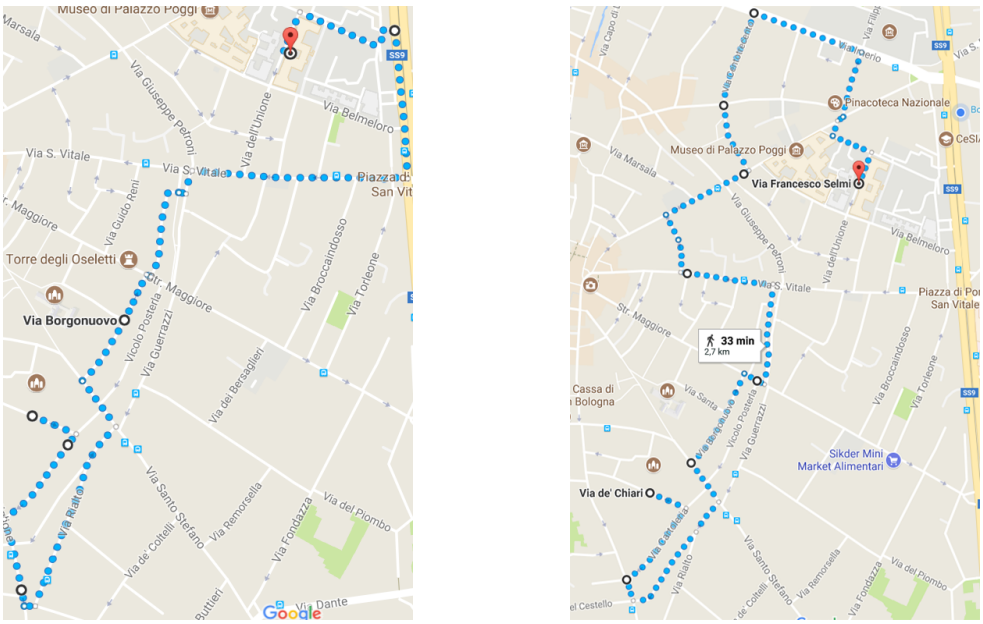

Fig. 2.5: Wei's routes

Sven

This morning Sven is going the Cineteca di Bologna to study some sources and will reach at noon his master thesis supervisor at the Department. The weather is not perfect, it is partially cloudy, but Sven prefers to use the bike because he will not have enough time for the Gym. So Sven prefers a longer path so as to do a good workout. Our PUMA proposes a longer path, a path through different green areas and some slopes, as shown in Figure 2.6.

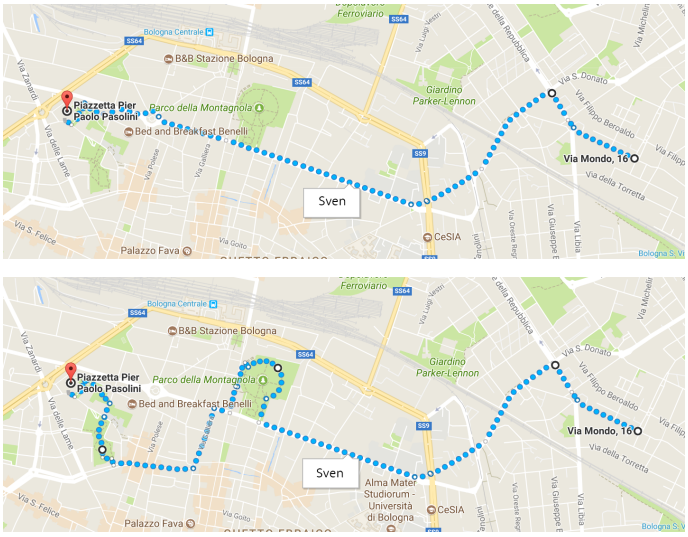

Fig. 2.6: Sven's routes

Elena

Elena is going to work, she will leave Tommy at the school on her way. The weather is very good, 25° C, a perfect spring day with a perfect temperature. She is leaving in at 7.45 AM, just in time to stop at the School, say bye to Tommy and go to work in schedule without being in a hurry. She decides to go safe using the available bicycle lanes and to select a route through parks and green areas to enjoy the spring weather and avoid a large exposure to pollutant. Our system proposes a path that does not cover Via Irnerio (as shown in Figure 2.7), a route not safe for cyclist because of the traffic, that includes cars and buses, and not clean, due to the pollution produced by these means of transport.

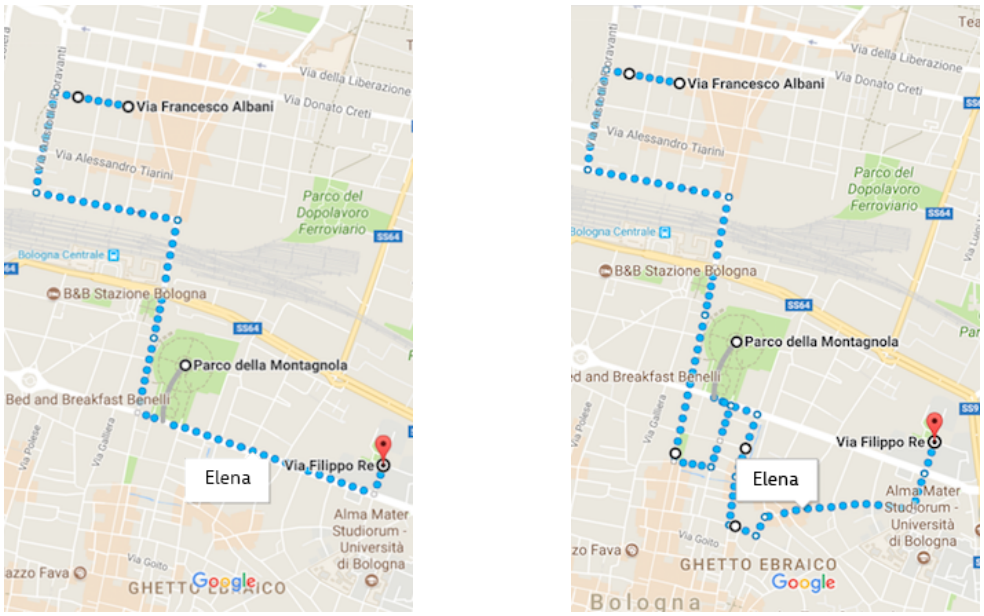

Fig. 2.7: Elena's routes

2.5 Conclusions

This chapter presents a Personal Urban Mobility Assistant (PUMA) system, based on the idea of letting citizens and public administrations gathering and exploiting integrated information about the urban environment, by using web applications and mobile devices. Users are at the same time producers and consumers of data and services, thanks to a cloud architecture approach, providing typical SaaS benefits. The chapter focuses on a specific mean of transport: bikes. It describes three different personas and related scenarios, with the aim of illustrating how our system can support smart and sustainable mobility in a urban scenario, thanks to crowdsourcing and crowdsensing activities.

Chapter 3

Canarin II: Designing a Smart e-Bike Eco-System

Mobility and ambient conditions are key factors in urban environments, affecting well-being and quality of life.
In this context, sensors, smart mobility, networks, connectivity can play a significant and strategic role, being exploited with the aim of improving data and information available to public administration and to each citizen. In this way, they can be supported in having more sustainable and aware behaviours and in getting useful information and services, improving their daily activities.
In this chapter, the project *Canarin II* is described along with the prototype of a smart bike eco-system; the e-Bike system was designed with the aim of collecting, aggregating and sharing data about air pollution and about the urban environment, which can be exploited in a smart mobility context thanks to sensor and vehicular networks.

3.1 Introduction

Air pollution plays an important role in human health and life quality, since it is responsible of different diseases, and it gains further importance in urban scenarios, considering that most people live in cities [34]. In fact, in 2016, the World Health Organization released a report about urban health, claiming that about 3.7 billion people live in cities today and that a further 1 billion will be added by 2030, with 90% of the growth being in low - and middle - income countries [35].

One of the most important and dangerous air pollutants is particulate matter [36, 37]. Particulate matter (PM), also known as particle pollution, is a complex mixture of extremely small particles and liquid droplets that get into the air. Once inhaled, these particles can affect the heart and lungs and they can cause serious effects. These particles come in many sizes and shapes and can be made up of hundreds of different chemicals. Some are emitted directly from a source, such as construction

sites, unpaved roads, fields, smokestacks or fires, but most of them come as a result of complex reactions of chemicals, such as sulfur dioxide and nitrogen oxides, which are pollutants emitted from power plants, industries and automobiles [38].

The traditional approaches to particulate measurement consist of complex, stationary, expensive equipment, and they provide significant limits to who can collect data and on how data can be accessed. Thus, they have not turned out to be adequate in the urban context, as particulate matter varies locally based on several factors including local pollution sources, which are typical of urban environments, such as cars.

In this context, new paradigms and technologies are emerging, which can be exploited and which can be combined and integrated, so as to provide an alternate and cost-efficient approach to PM collection. In particular, in the last years, the Internet of Things has gained increasing attention from people all around the world. The basic idea of this concept is the pervasive presence around the users of a variety of things or smart objects - such as Radio-Frequency IDentification (RFID) tags, sensors, actuators, mobile phones, etc. - which are able to interact with each other and to cooperate with their neighbors to reach common goals, through unique addressing schemes [39]. Thus, the paradigm of particulate measurement is now changing thanks to the IoT and to the new possibility of using lower-cost, portable pollution sensors, allowing the collection of real-time data that can be exploited in many and different contexts. This new family of sensors provide new avenues to air pollution monitoring applications that directly involve citizens [40].

All these new applications can be categorized as related to a *smart environment*. The core concept of *smart environment* is the use of technology to increase sustainability and to better manage natural resources [19]. Smart environment is one of the dimensions that define a smart city, which is well-recognized as "a city well performing in a forward-looking way in economy, people, governance, mobility, environment, and living, built on the 'smart' combination of endowments and activities of self-decisive, independent and aware citizens" [41].

Another dimension of a smart city that could be interesting in this context is *smart mobility* [16]. All these pollution data could be used in different ways with the aim of supporting smart mobility and citizens while they move across the urban environment. A first example is the integration of data about pollution in a travel planner (i.e. Open Trip Planner), so that it could take into account them while planning pedestrian paths [20]. A second example could be the use of these data in mobile applications, so as to let the users check their exposure to air pollution while they walk or they ride a bike.

In this context, bikes represent a very interesting vehicle, because they could play a key role in urban transportation scenarios [26]. They don't cause pollution, hence they can be exploited as a vehicle which can be equipped with pollution sensors, without affecting data collection.

Adams et al. [42] described three new different typologies of bikes which could be potentially revolutionary:

- *Electric bikes (or e-bikes)*: electric, pedal-assist bike, with a power-on-demand basis.
- *Connected bikes*: bikes that can communicate via mobile phones, which can be wireless connected or wearable integrated.
- *Smart bikes*: bikes equipped with sensors that can communicate via mobile phones or wireless connected.

In this chapter we present a smart e-bike prototype, equipped with a system devoted to collect data about particulate matter and to elaborate and share such data. In particular, a prototype of Canarin II (a sensor for detecting particulate matter, as a pollution monitor [14]) has been installed on an electric bike, which is connected to the user's smartphone, so as to support data collection and sharing. The purpose is to collect pollutant data while the user rides his/her bicycle to map pollution into the urban context and to let the elaboration of the gathered data, which are transmitted to a server. Some preliminary experiments have been conducted and here presented, thanks to the collaboration among the University of Bologna, the Universite Pierre et Marie Curie, the Macao Polytechnic Institute, and the Asian Institute of Technology. The remainder of the chapter is structured as follows. Section II briefly describes the background, by presenting some related work. Section III summarizes the main issues that have been taken into account while designing the whole smart e-bike ecosystem. Section IV presents the architecture we have defined, while the prototype we have built and tests are illustrated in Section V. Finally, Section VI concludes the chapter highlighting some final remarks and future work.

3.2 Background and Related Work

In this section, we briefly introduce some work related to our prototype, presenting some cases based on the collection of pollutant data in urban scenarios.

New low-cost sensors are at the basis of several studies. White et al.[43] created a PM monitor made by micro-fabrication techniques, derived from the manufacture of integrated circuits, that can be accessed directly by cell phone technology. Mead et al. [44] have used miniature, low-cost electrochemical gas sensors for sensing at parts-per-billion (ppb) for gases relevant to urban air quality. They used low-cost and highly portable sensor, thus allowing the deployment of scalable high-density air quality sensor networks at fine spatial and temporal scales, and in both static and mobile configurations. Zampolli et al. [45] developed a miniaturized, low-cost electronic nose based on state-of-the-art metal oxide sensor. The proposed device was targeted to the quantification of carbon monoxide and nitrogen dioxide in mixtures with relative humidity and volatile organic compounds by using an optimized gas sensor array and highly effective pattern recognition techniques. Tran et al.[46] proposed a novel battery-free sensor module which has been used to measure the

concentration of volatile organic compounds, ambient temperature, relative humidity, and atmospheric pressure for monitoring air quality in indoor environment.

In general, proposed systems for environment monitoring mainly follow two different approaches. A first approach is based on building a network of sensors devoted to monitor indoor and outdoor air quality. Studies based on this approach analyze the architecture of the network of sensors [47], taking into account different problems, such as consistency and sensitivity of a sensor to its micro-environment local conditions [48]. The second approach is based on the use of mobile sensors with the aim of monitoring air quality in different parts of a city concurrently and continuously [25]. Such an approach can be applied in different ways, ranging from the the use mobile sensors unit [49], to the use of sensors for smart phones [50] or for wearable devices [51].

One of the most commonly used approach to data collection, given mobile low-cost sensors, is crowdsensing, which can be exploited in very efficient way in different contexts [52]. Woodruff2010 et al.[53], in the Common Sense project, developed a website that shows a unique aggregated evaluation of the pollutants measured by a set of sensors owned by the members of a community. Data coming from such sensors have been normalized over time to the EPA's Air Quality Index. All of the views on the site use color encodings and descriptors to communicate data and results, because usually citizens have no confidence with raw pollutant concentrations. Ziftci et al.[54] developed Citisense, a system that allows users to collect air quality data via portable sensors and to get real-time feedbacks about pollution with an Android application or view summary information in a map. Kuznetsov et al.[55] proposed a modular system of low-cost and networked sensors that measure environmental factors such as air pollution, radiation, water quality or noise. Aberer et al.[56] presented OpenSense, an open platform to monitor air pollution using wireless and mobile sensors by adopting complex utility driven approaches. They installed these sensors on vehicles of public transports, such as buses and trams. Hasenfratz et al.[50] built GasMobile, a prototype system for participatory air pollution monitoring. They used a small, low-cost, and on-the-shelf sensor with the aim of monitoring the ozone concentration. An Android application has been provided in order to let the user exploit and browse the real-time collected data.

In this context, gamification elements and techniques can provide great benefits, obtaining the successful effect of involving people in the mapping of urban environments in order to have full data coverage [57]. A similar approach was proposed by Garcia et al. [58], where the authors developed mobile applications with the aim of monitoring noise pollution. They proposed two different applications, targeting two user profiles, Achiever and Explorer. In the first case, they provide a system where the user can win points and conquer areas, with the purpose of gathering as many measurements as possible. While, in the second application, they gave the user the possibility of collecting data around the city, with the purpose of getting trustworthy and reliable observations.

With the availability of all these data about pollution, the use of machine learning

algorithm can be very interesting. A first example is the use of a neural network to predict the category of the current pollution, chosen between predefined pollution source categories, such as traffic, urban and photochemical pollutions, as proposed in [59]. The authors of [60] proposed the use of an artificial neural network in environmental sensing and measurement systems with the purpose of compensating for the effects of influence quantities. Another example of machine learning approach applied to pollution sensing is presented in [61], where sub-micron sized ambient air pollutants are predicted by means of machine learning algorithms.

Since data collection is a significant time consuming activity, it is interesting taking into account how and when it can be done. It is strategic letting a wide range of users conduct it as a daily routing. For this reason, many proposals are based on the use of bicycles, transforming such means of transport in mobile sensors [62]. Kanjo et al.[63] built a cycling courier with a data logger, equipped with a pollution and a noise spy. They use such data to map pollution, in terms of both air and noise ones, through the Google Earth interface.

All these elements and experiences have been taking into account in defining the bike eco-system we propose, and they are at the basis of the design issues that have driven our experiment, as described in the following sections.

3.3 Design Issues

In this context, we have designed a smart bike eco-system, with the aim of collecting data about smart mobility in a urban environment with a crowdsensing and crowdsourcing approach. Such information can be aggregated and integrated with other data coming from other types of sources (i.e. data coming from other sensors, open data coming from municipalities or other public administration and services, etc.), obtaining a data platform. This could support citizens, equipping them with services and information useful while they move across the city by means of different vehicles and transport systems. A smart bike eco-system should take into account many and different issues, but we would like to drive a special attention to the following ones:

3.3.1 Health

Low-cost sensors can be installed on the smart bike, with the aim of collecting data about pollution in the urban environment, while the users ride their bikes. Moreover, the bike eco-system should be equipped with GPS or with geolocation sensors, in order to add information about position to the sensed data, together with the sampling time. This forms a sensor and vehicular network [27], devoted to gather geolocalized data, enriching the city map with a specific layer of information, on a server side [33]. Pollutant sensors could communicate such data through the city-wide wireless network or through the riders' smartphones, exploiting heterogeneous

networks [29].

Users are simultaneously data producers and data consumers, because they could exploit specific services, which can be tailored on their needs, while they ride the bike. For instance, based on the collected data, a mobile application could plan bike routes across the city, taking into account specific levels of air pollution, avoiding the most dangerous areas, including the most healthy ones.

3.3.2 Wellness

Smartphone sensors and sensors mounted on the bike can be exploited so as to gather information about the rider's workout, in terms of physical exercise. Accelerometers, bicycle tachometers, heart rate monitors, etc. can provide interesting and useful data in this sense. Travel planner can be used in order to support the estimation of the calories which will be consumed while riding a proposed route. Commonly used APIs (i.e. Google Maps Road) can play an interesting role in reconstructing an already ridden route, and hence computing and providing data related to the conducted physical activity. Other characteristics can be taken into account in contexts related to riders' wellness, such as steepness, pleasantness of the panorama and sights during the route, posture while riding (which can be monitored by wearable sensors).

The bike eco-system, together with mobile apps available through the user's smart phone, can contribute in proposing bicycle routes meeting the rider's needs in terms of workout, right posture, pleasantness, entertainment [64], etc.

3.3.3 Safeness

The smart bike eco-system can play a strategic role in collecting and providing information about safeness and security of the rider and of the bike, too. Accelerometers and geolocation sensors can record data about bike falls; while the riders can add details and share information about street traffic and dangerous roads thank to cameras, crowdsourcing mobile apps and to open data which would come from municipalities, public transport agencies and companies [65]. Such data sources could provide also information about urban security (in terms of offense against the person), in specific areas of the city [66].

All this information can enrich the available data at the basis of the computation of personalized paths, hence supporting the users in getting a safer route in terms of avoiding dangerous areas and roads.

In this context, GPS and anti-theft sensors can represent a mean to protect the smart bike itself, preventing stealing or providing support in finding a stolen bike.

3.4 Our Prototype Architecture

Our prototype of smart bike eco-system is based on the following elements:

- An electric bike (thanks to the collaboration of the Mobility Manager Office of the University of Bologna).
- A kit equipped with different sensors (named Canarin [14], designed and developed thanks to the collaboration among the Macao Polytechnic Institute, the Asian Institute of Technology, and the Pierre and Marie Curie Sorbonne University). The kit sensors (shown in Fig. 3.1) has been installed on the e-bike (as depicted in Fig. 3.2).
- A smartphone, acting as an access point from the bike sensors to the Internet.
- A web-based application, showing data collected during bike rides.

In particular, we have exploited the Canarin 2.0 architecture, which is based on its previous version 1.0[14] and now relies on a UDOO Neo Full, an Arduino-powered Android/Linux single board computer. This new version aims to overcome some limitations emerged in Canarin 1.0, as follows:

- *Storage extension*: the new board can save the PM sampling in an external SD card. This allows to store a huge quantity according to the SD card capacity.
- *WiFi enhancement*: the new network card allows a better connection between the node and the router. It also allows WPA2-Enterprise connection, such as Eduroam in universities.
- *WiFi optimization*: the board collects data even without the need of a stable connection. It stores the GPS location and the date time of the sampling; eventually it sends such data to the server when a connection has been established;
- *Sensors enhancement*: an improved PM sensor has been added. It can collect PM_1 and formaldehyde as well in addition to the $PM_{2.5}$ and PM_{10} values. A more accurate temperature and humidity sensor has also been installed, together with a sensor which can gather values of UVI (ultra violet index).
- *The microprocessor*: the board runs thanks to a 1GHz ARM Cortex-A9 microprocessor. The use of Linux guarantee a full control of the board potential, as well as flexibility and reliability.

Due to the power of the network card and the microprocessor, the battery usage also increases: the multiple day system duration of Canarin 1.0 decreases to one day, depending on the battery capacity.

The new system architecture is therefore structured around two layers: all the sensors runs on a Arduino UNO-compatible platform that clocks at 200 MHz, based on a Cortex-M4 I/O real-time co-processor, while a Linux based OS stores the data into files and it establishes a connection to the server in order to send data. The sensor design is based on an in-house PCB that hosts the board and the sensors welded. The communication is based on WiFi and the board can also connect to EAP-SIM networks by means of a USB SIM reader, as Free-Telecom WiFi in Paris. The communication protocol is based on UDP, as well as the Canarin 1.0 protocol and it communicates to an enhanced MySQL database version. The others sensors have remained the same: the system collects not only particulate matters, but also temperature, relative humidity and air pressure.

We also designed an in-house 3D printed PLA box has been also designed to wrap the battery and the PCB together (shown in 3.1). Everything fits in a 19x15x7 cm and it weights about 900g due to the battery and the enclosure; a smaller and more portable 3D printed box is about to be produced.

Fig. 3.1: Canarin II - version 1

3.5 Prototype Presentation

The prototype of our smart bike eco-system has been preliminarily tested in the urban environment of Bologna (Italy), during May and June 2017. Tests have been conducted by riding our e-bike in different areas of the city, in different lanes and streets (from quite bike lanes in parks, to traffic main streets, from alleys in the

historical down town to main roads linking the city center to the industrial areas). Overall, these tests took 4 hours and covered 40 kilometers in the Bologna urban areas, as average per day. Data about the tests were stored on the server-side application, which can be accessed through a web browser.

Fig. 3.2: Our Smart Bike prototype

Samples were taken every 40 seconds, and for each sample we have collected the following data: date and time, GPS coordinates, temperature, humidity, air pressure, and values of formaldehyde, PM_1, $PM_{2.5}$, and PM_{10}.

The samples collected during three of the bike ridden routes are shown in Figure 3.3 (a specific color identifies a specific route), which is a screenshot taken from our web application. Here, each marker represents a sample; the user can click it to access details and values related to that sample.

Our web application is based on the following APIs:

- Google Maps Javascript API, to show a map in which air pollution data, collected by the sensor, is displayed.
- HTML5 Server-Sent Events, to allow the web page to get updates automatically from a server, so there's no need to reload the page to show new data.

Furthermore, server side, we developed a PHP script to read data from the MySQL database: it checks if the data read has been already sent to the client and, if not, it returns them in JSON format. When the web page retrieves new data, a JavaScript function is called in order to create a new marker for each sampling that is then added to the map. An event listener is added to each marker and, when a user clicks on a marker, an info window is shown. This modal popup contains all the data about the stored sample, including its date-time, PM_1 value, $PM_{2.5}$ value, PM_{10} value, temperature, relative humidity and air pressure.

This web application allows users to monitor their exposure to pollutants in all their movements in their daily routine. Eventually, they could decide to change their habit preferring a healthier route.

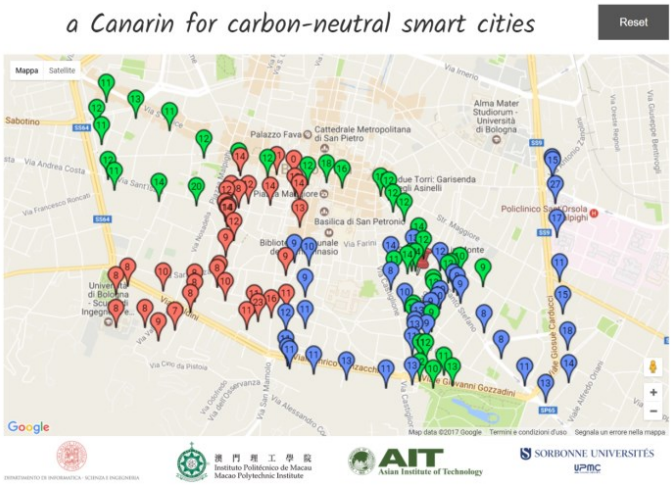

Fig. 3.3: Frontend

On the 6th of June 2017, on the occasion of G7 Ministerial Meeting on Environment, we presented a demonstration of our prototype of smart e-bike eco-system. One of the authors rode the bike in a circular route, so as to let the audience attend the start and the stop and to enjoy details about the data collection during the ride, thanks to the web-based application.

3.6 Conclusions

This chapter aims to present a prototype of smart bike eco-system we have designed and developed. At this first stage, we have payed specific attention of the idea of creating a sensor vehicular networks based on bikes, so as to collect, aggregate and share data about air pollution in urban environment. Hence, we have equipped our bike with a kit of sensors devoted to sample values of PM, together with data and details about environment condition (i.e. air pressure, temperature, humiddity, etc.). We have conducted some preliminary tests and we have successfully presented a demonstration of our smart bike eco-system on the occasion of the G7 Ministerial Meeting on Environment.
This is a preliminary work, which can be considered as a starting point for several considerations and plenty future work, from a wide range of points of view, from quality and quantity of collected data, to data trustworthiness, from connectivity issues to power required for the sensors eco-system, from elements about user interaction and interface to user modeling and profiling, just to cite few of them. We have planned a more extensive test campaign, which will involve more bikes equipped with our kit of sensors, providing our mobile apps to the riders, with the aim of drawing some initial considerations about the vehicular network aspects of our idea.

Chapter 4

Canarin Nano: a wearable pollution monitor for the allergic rhinitis and sleep quality case study

In the last decade, studies that assess the harmfulness of atmospheric pollutants[67, 36] have offered increasingly alarming results, despite the reduction of certain important pollutants. Defining the impact of these pollutants on human health is challenging without considering all multi-environmental variables.
A solution to this can be offered by customized measurement wearables, which are promising tools for quantifying the exposure to the air pollution. The technical progress made in recent years in the fields of electronics and instrumentation, in fact, enabled the creation of new geo-localized mobile pollution sensors that are cheaper and more compact than the reference stations used so far.
Canarin Nano reimagines the old and static Canarin, described in the chapter 3, in a wearable option.

Fig. 4.1: Canarin NANO - Wearable

4.1 The context: the POLLAR clinical study

The need to further reduce the Canarin II large design came from the POLLAR[1] clinical study done in Grenoble (France) from March 2019 to January 2020. A prospective monocentric open study entitled *Impact of pollution on allergic rhinitis and sleep quality: the POLLAR study*[68], including 52 patients, was performed by Grenoble Alpes University (UGA) in collaboration with the Grenoble Alpes University Hospital (CHUGA). The main goal of the clinical study was to evaluate the impact of pollution and exposure to pollens on sleep parameters in patients with or without allergic rhinitis (AR).

Participants were equipped with different portable sensors for air quality measurement (particle pollution, gas, etc.), a Global Positioning System (GPS) module, a temperature and humidity detector, and physical activity sensors. A proper measurement campaign was carried out over eight periods of one week, both in winter and in summer.

The data obtained by Canarin Nano were also studied in cooperation with CEREMA[2] (Centre d'etudes et d'expertise sur les risques, l'environnement, la mobilité et l'aménagement) and INSERM[3] (Institut national de la santé et de la recherche médicale), from an epidemiological perspective. The results of the study were also provided to the POLLAR team, who used them as materials for modeling work, and for the development of reliable solutions. This enabled them to improve the performance of their analytical and numerical models.

4.2 Design

Although the Canarin II partially satisfied the requirements set by the POLLAR team, some size and weight adjustments were necessary to make the Canarin a proper wearable sensor device. In addition, in order to track the patients outside their homes, we opted for a small cellular platform to orchestrate all sensors and the network connectivity. After evaluating various IoT market solutions, we chose the Particle Electron[4].

4.2.1 Particle Electron

The Electron is a tiny development kit manufactured by Particle and employed to create cellular-connected electronics projects; it is equipped with an embedded U-blox SARAU260/U270 for 2G/3G connection that allows users to rely on a secure and stable network. In details, the platform comes with

[1] `https://eithealth.eu/project/pollar/`
[2] `https://www.cerema.fr/fr/mots-cles/paris`
[3] `https://www.inserm.fr/`
[4] `https://www.particle.io/`

- U-blox SARA-U260/U270 (3G with 2G fallback)
- STM32F205RGT6 120MHz ARM Cortex M3 microcontroller
- 1MB flash, 128KB RAM
- BQ24195 power management unit and battery charger
- MAX17043 fuel gauge
- RGB status LED
- 30 mixed-signal GPIO and advanced peripherals
- Open source design
- Real-time operation system (RTOS)
- FCC, CE and IC certified

Even if Particle offers its own phone sim card for cellular connecting and data uploading to their dedicated cloud storage, we preferred, instead, a third-party sim provided by Thingsmobile[5], relying on our server.
Furthermore, we improved the sampling capability by adding small and powerful new sensing chips: (i) the new BME680 gas sensor made by Bosch, which integrates high-linearity and high-accuracy gas, pressure, humidity and temperature sensors. It can also detect a broad range of gases such as volatile organic compounds (VOC); (ii) the Sensirion SGP30, a digital multi-pixel gas sensor platform, which samples total VOC (ppb) and H_2-based CO_2eq (ppm); (iii) a smaller version of the PM sensor used in the Canarin II, a light scattering optical analyzer to count PM_1, $PM_{2.5}$ and PM_{10}.
To increase Electron's default storage capacity, we added a Winbond Electronics 8MB SPI NOR flash memory that can store up to 48 days of samples offline, in case of connectivity issues.
The whole platform is powered by a lithium polymer 4000 mAh battery placed under the printed circuit board; it can be recharged with a 5V 2A AC/DC charger in 4/5 hours while the Canarin Nano can operate for up to 10 hours.
Due to privacy reasons, the GPS module was disabled for the POLLAR project. Even the code was re-designed from scratch. Previously developed for a NXP i.MX 6 SoloX inside the UDOO Neo, the current Arduino-like sketch is coded in the Particle Electron web IDE; the sketch is then exported into a binary file and flashed into the Canarin Nano at a later time using Particle command line controls. The sampling algorithm flow is similar to the Canarin II's one, but we exploited the Electron multithread capabilities to separate the connection control from the sensors management as described in fig. 4.3 at the end of this chapter.

[5] `https://www.thingsmobile.com/`

4.2.2 Remote control

The remote control was a brand-new implementation that introduced several setting customizations. The user can now dynamically change various parameters without flashing a new firmware; in details the user can set

- Interval time, i.e. the sampling interval between two data; (Default: 1 minute)
- Cloud Mode, which is Boolean value: if true, the Canarin Nano disconnects from our server to connect to the Particle Cloud at the next loop; there the user can check vitals like signal strength, memory usage, and signal quality; (Default: false)
- Cloud time timeout, i.e. how many seconds the Canarin Nano has to stay connected to the Particle Cloud; once connected, the Canarin Nano suspends the internal sampling; (Default: 120 seconds)
- PM sampling time, i.e. how many seconds the PM sensor works in the time interval; (Default: 30 seconds)
- Modem reset, which reboots the modem connection; (Default: false)
- Sleep time, i.e. how many seconds the Canarin Nano goes to deep sleep between the samples;
- Battery Sleep time, i.e. how many minutes the Canarin Nano goes to deep sleep if the battery is very low; (Default: 30 min)
- NTP time, i.e. if the Canarin Nano has to ask the exact datetime to a NTP server; (Default: true)
- Exact minute, i.e. if the sampling time must start at MIN:00; (Default: true)
- File system erase, which removes all offline data stored in the flash memory; (Default: false)
- Default settings restore: if true, it resets all these variables to the default value. (Default: false)

4.3 Implementation

The Canarin Nano's workflow is coded as a multithreading Arduino-like sketch: (I) one main thread, i.e. the loop, is the sensors orchestrator; it saves all the values from the sensors, ready for uploading; if the platform is offline, it saves the samples in the internal flash 8MB memory; (II) the second thread keeps the cellular connection alive, verifying its status and signal strength every minute; in

case of multiple server timeouts or a failure, the connection thread may reboot the modem to reinitialize it for a new cellular handshake; (III) the last thread checks the GPS satellite signal every second and updates the latitude, the longitude and the altitude of the next sample.
In the setup, we set every sensor up, along with the Particle Electron's interfaces. In case the SPI NOR flash memory has not been initialized before, the setup may take up to 5 minutes to format it as SPIFFS[6] (SPI Flash File System). This happens only once per platform.
Once the flash memory is mounted, a file is reserved for the default parameters described above. A stored file is necessary to keep the settings saved in case the platform may shutdown.
Once the flash memory is mounted, a file is reserved for the default parameters described above. A stored file is necessary to keep the settings saved in case the platform may shutdown. Next, the setup writes the sim card APN settings in the internal modem; the APN can be easily changed by flashing a new firmware; the unique ID number is set using the Particle Electron's serial number; eventually the loop, the connecting thread and the GPS thread start.

Algorithm 1: Canarin Nano SETUP()

```
1:  Serial initialization(9600);                          {Serial Print Debug}
2:  Serial1 initialization(GPS_BAUD,SERIAL_8N1);          {GPS init}
3:  Serial5 initialization(9600);                         {PM sensor init}
4:  Wire initialization();                                {I2C sensors init}
5:  BME680 begin();
6:  SGP30 begin();
7:  spiFlash begin();                                     {Flash memory init}
8:  spiFormatFS(8MB);                                     {Flash format check and init to 8MB}
9:  spiMountFS();
10: createDefaultVariables();                             {Create one file in FS for settings}
11: Particle.keepAlive(600);                              {Avoid cellular disconnections}
12: apnHelper.setCredentials();                           {Set APN}
13: getID();                                              {Create node number identification}
14: t1Thread = new Thread("connectingT", connecting, NULL);
15: t2Thread = new Thread("gpsT", getGPS, NULL);
```

At the beginning, the loop checks the battery status and puts the system into a *Deep sleep* if the lithium polymer 4000 mAh battery voltage is below 3.5V; this is necessary as battery undervoltage protection. A red warning led turns on while the whole system shuts down. The user must charge the Canarin Nano or turn the general switch off, otherwise the system checks if the voltage is above the minimum

[6] `https://github.com/pellepl/spiffs`

level every *Battery_Sleep_time* cycle (30min by default).
After that, remote settings are checked, such as the default variable restore, the file system erase or the modem reset. If the three values are false, the sampling part begins: one by one, the battery level and voltage, the air temperature, humidity and pressure, the volatile organic compounds, the H_2-based CO_2 equivalent, PM_1, $PM_{2.5}$, and PM_{10}.
As a final step, the message is built following the protocol:

$$\{SENDING\_CODE, NODE\_ID, PROJECT\_ID, [PAYLOAD]\}$$

The sending code and the payload may vary depending on the scenario:

- **0: INSERT NEW NODE**
  $MSG : \{0, NODE\_ID, PROJECT\_ID, [NODE\_NAME]\}$
  The new Canarin adds itself to the cloud. This usually happens at the very first loop; a node may add a custom name as payload. By default "N"$+NODE\_ID$ is used (e.g. "N12345")
  The server checks if the sensor nodeID is already registered in another project id; if so, the server sends an ACK back to the sensor with the current project id in the payload.
  RETURN CODE: 0 (error) or 1 (ok)

- **1: INSERT A NEW SAMPLE**
  $MSG : \{1, NODE\_ID, PROJECT\_ID, TIMESTAMP, [(VALUE\_ID1 : VALUE\_NUM), (VALUE\_ID2 : "VALUE\_STRING")...]\}$
  The Canarin, at the end of the sampling loop, sends a packet to the server with all sensors values. According to the protocol, every variable has a pre-defined ID number (e.g. 4:temperature, 5:pressure, 8:$PM_{2.5}$ etc.). These tuples are defined in the Canarin Database "unified-type" table
  Example: {1, 12345, 42, 1544858269, 1:0, 2:0, 3:0, 4:29.65, 5:22.03, 6:1008.24, 7:10, 8:30, 9:10, 10:4.03, 11:"HELLO WORLD" }
  RETURN CODE 0 (error) or 1 (ok) or 2 (new settings are available from the cloud)

- **2: SET NEW SETTINGS**
  $MSG : \{2, NODE\_ID, PROJECT\_ID, [SET\_ID : SET\_VALUE, ...]\}$
  The node sets the new settings server-side. A similar ID/VALUE tuple is pre-defined as with the samples values above (e.g. 0:Interval_time, 1:Cloud_Mode, ... , 5:Modem_reset etc.). These tuples are defined in the Canarin Database "unified-conf" table
  RETURN CODE 0 (if error) OR 2 (ok)

- **9: DELETE/MOVE NODE**
  $MSG : \{9, NODE\_ID, PROJECT\_ID, [NEW\_PROJECT\_ID, NODE\_NAME]\}$
  With this command, the node can either delete itself from the project if payload is NULL, or move to another project.
  RETURN CODE 0 (if error) OR 1 (ok)

The couple $NODE\_ID$ and $TIMESTAMP$ is used as MySQL key in the data table. The timestamp is the epoch of the sample, i.e. how many seconds have passed since January 1, 1970 00:00:00 UTC. $PROJECT\_ID$ is used server-side to redirect the values to the correct data table. At the moment we have 10 active projects that belong to the Canarin Project; the POLLAR project in Grenoble is one of them. Once the message is ready for sending, sendSample() checks the connection status and it sends it accordingly. If the connection is down, the sample is queued in a file; once the Nano comes back online, every saved sample is uploaded with the FIFO approach. If the connection is stable, the Canarin may also connect to the Particle Cloud, ready for On-The-Air (OTA) updates or vitals check, and synchronize the internal clock with the Network Time Protocol (NTP) server. WaitForNextSample() waits for the offset between the end of sampling and the next minute, then the following sensing loop restarts. The Canarin may either enter deep sleep, switching sensors off, or soft sleep, keeping sensors calibrated for better accuracy.

Algorithm 2: Canarin Nano LOOP()

```
    if BatteryCheck() then
2:      System.sleep(batterySleepTime);
    end if
4:  if restoreVars then
        restoreVarsToDefault();
6:  end if
    if erase_FS then
8:      fs.erase();
    end if
10: if resetModem then
        resetModem();
12: end if
    getBatteryVoltage();
14: getBME680Values();
    getSGP30Values();
16: getPMValues();
    epochNow = getNow();
18: msgToSend = "{SEND_COMMAND, nodeID, ProjectID, epochNow,
    sampleValues}"
    sendSample(msgToSend);
20: waitForNextSample();
```

Algorithm 3: Canarin Nano sendSample(msg)

```
    if Cellular.ready() then
      if ParticleCloudMode then
3:      connectToCloudMode();
      end if
      if getNTPTime then
6:      getNTP();
      end if
      if firstSample then
9:      insertNodeIDtoProject();
      end if
      uploadSample(msgToSend);
12: else
      saveSampleOffline(msgToSend);
    end if
```

4.3.1 Server side and frontend

On the server side, different Docker[69] containers are running and serving different Canarin projects flows. Every container runs a python script that is responsible for demultiplexing every incoming message. It checks the correct match between $Project\_ID$ and registered nodes, and then it stores the values in a single row in the dedicated data table.
The server also updates every Canarin when they need to update their settings, i.e. a correct message with SENDING_CODE 2 was sent.
The server is an Amazon AWS EC2 *t2.micro* instance based in Frankfurt. The instance runs on a Linux Ubuntu 14.04.6 LTS with a 3.13.0-170-generic Linux kernel. The *t2.micro* machine has one vCPU and 1GiB RAM, 32GiB SSD, that - so far - is enough for 100+ Nanos scattered throughout the world.

On the database side, we rely on Amazon AWS services too. Two Amazon Aurora instances make up a regional cluster based in Frankfurt. Two *db.t2.small* instances offer 1 vCPU core and 2GiB RAM, with low-medium throughput. One reader and one writer instance work together to prevent DDOS attacks and make fall-back recovery available. Here our MySQL Database runs on 5.6.mysql_aurora.1.22.2 engine.

On the web side[7], we improved our user-friendly frontend to enable users to observe and download data from the nodes. 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. For future reference, e.g. for later big data analysis, we added the 'Download data' section that generates a custom CSV file from old data. Here the

[7] `https://canarin.net/`

user can select different time windows, filtering on a set of Canarin and a set of sensors.

4.3.2 Hardware design

Thanks to the experience gained with previous Canarin models, we designed a new light Printed Circuit Board to avoid using wires and to reduce the Canarin's overall dimension. They are printed with FR4 TG150, a glass-reinforced epoxy laminate material; the size is 51.5 mm x 93 mm x 1.2 mm.

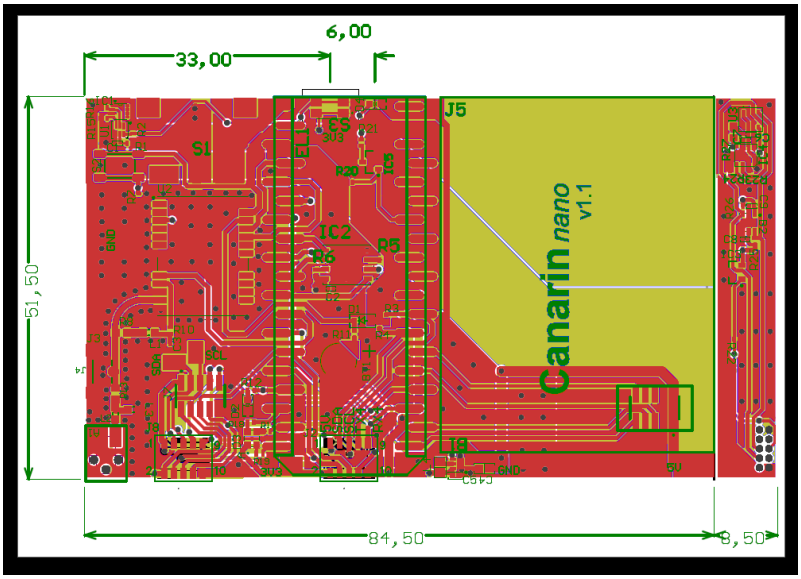

Fig. 4.2: Canarin NANO PCB

In order to power the PCB, two copper plates are placed underneath. These plates are directly connected with the two copper tabs that protrude from the lithium battery. This connection has the dual function of powering and recharging if the Particle Nano is connected via USB and the power switch is on at the same time.

Along with the PCB design, we also built a new 3D printed polylactic acid (PLA) box to tangle the PCB, the sensors, the antennas and the 4000mAh lithium battery together. Its overall size is 9.65 x 6.24 x 3.19 cm and it weighs about 177gr. Compared to the Canarin II (version 2 described in section 5.6), we ultimately managed to reduce the weight by 1/9.

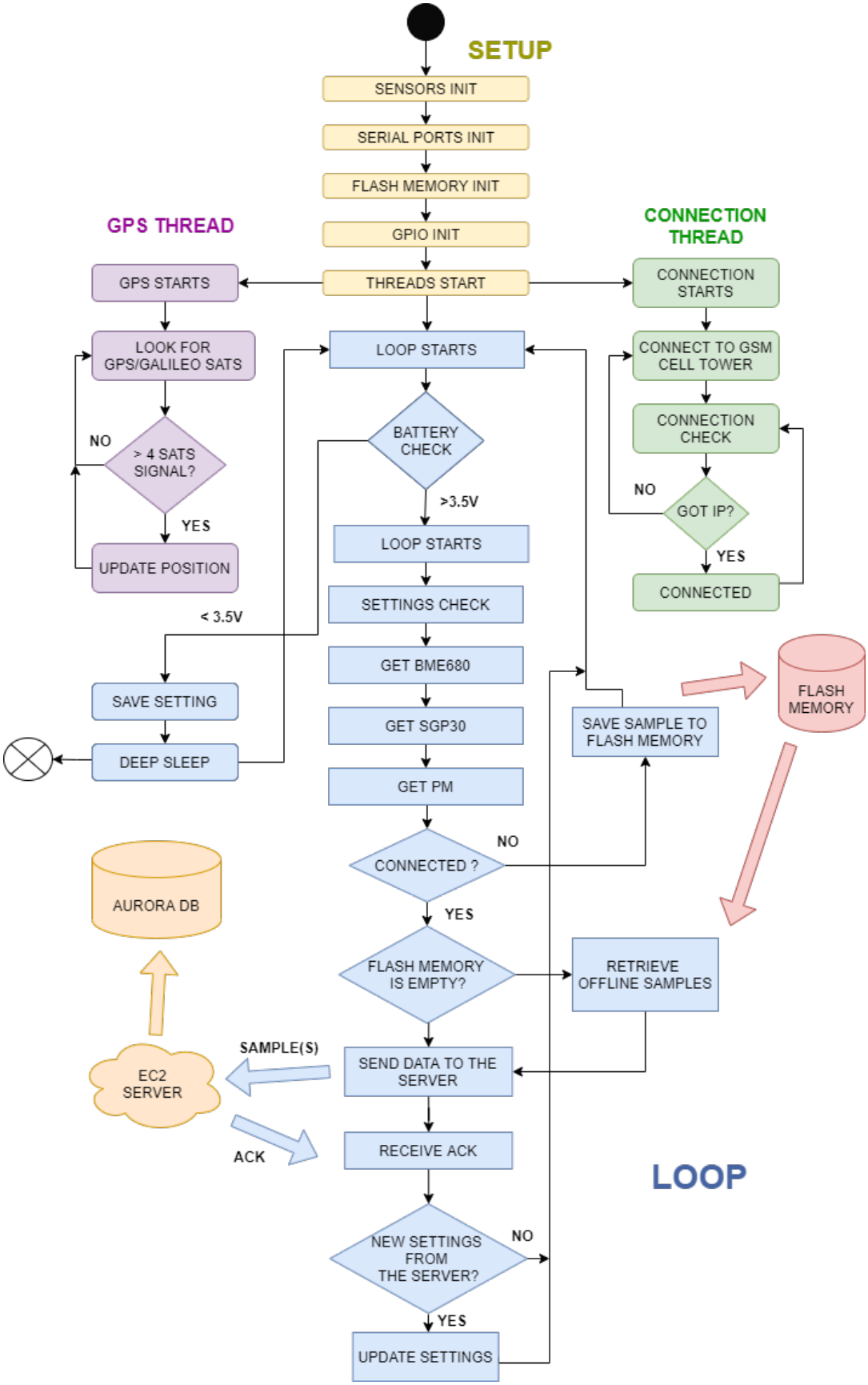

Fig. 4.3: Canarin NANO Flowchart

Chapter 5

Monitoring cultural heritage buildings via low-cost edge computing/sensing platforms: the Biblioteca Joanina de Coimbra case study

Climate change and higher level of pollutants are affecting our heritage affect the conservation abilities and endangering centuries old buildings and artifacts. Governments and conservation agencies alike are scrambling to monitor and prevent disasters that would result in the loss of precious cultural artifacts. Sensing technologies, on one side, and new restoration techniques on the other, are in being used in this effort however they result in high costs and complex installation logistics.
In this chapter we present the first of the two key sustainability C-Continuum macroscenarios: a pollution monitoring study for the *Biblioteca Joanina*, a baroque library situated in University of Coimbra, that has been inducted in the UNESCO world's heritage list in 2013.

In contrast with the classic practices involving noisy and obtrusive sensing stations, the study has been conducted using more than ten Canarin II platforms; after the e-Bike experimented illustrated in chapter 3, we further improved the Canarin, now more solid, smaller, and more discreet. As it was for the previous version, even this one was built in a joint effort by the Sorbonne Université, the Macao Polytechnic Institute and the Asian Institute of Technology (Thailand). To date (November 2020) the total samples collected in the library are more than 10 and a half million. Our study shows how is possible to use a low-cost platform for cultural heritage monitoring and preservation, how it can support decision-makers in execute simple policy changes, and yet achieve substantial impacts in the preservation efforts.

5.1 Introduction

After the introduction of Decree Law n. 78/2006 April 4, in Portugal, the indoor air quality (AIQ) has become a social need and national buildings, offices, libraries and shops started to tackle the airborne pollutants problem. Some sites in particular, such as old libraries and ancient museums need further attentions to manage 'the invisible killer', as has been coined the air pollution by the World Health Organization (WHO) [70], assuring the visitor an healthy location and guaranteeing the correct preservation of works of art and cultural heritage [71].
This represents the first action to manage a phenomenon in a different context than industrial and urban ones.
Several invasive techniques are available in modern building such as air conditioning (HVAC) and heating monitoring, but different strategies must be implemented in historical sites where their structure is part of the heritage, maintaining contaminants levels at low concentrations for human health [72]. Since most museums or libraries cannot afford dedicated complex monitoring systems, the Internet of Things (IoT) represents a promising, cost-efficient, not-invasive approach to measure air pollutants.[40, 73]
In this context, in June 2017 we started a monitoring campaign into the XVIII century Baroque library of the University of Coimbra - also called Joanina Library (Portuguese: Biblioteca Joanina) - placing several our sensor stations inside the building. From 2013, the historic centre of the city, including the University of Coimbra and its library, were declared a World Heritage site by UNESCO [74] making them one of the most visiting sites in Portugal drawing the attention of local and foreign tourists.
In particular nine Canarin II (a sensor for detecting particulate matter, as a pollution monitor [14]) have been installed in the three floors of the library, gathering different parameters 24/7 providing a collection set of data, useful by third parties to assess their own air pollution models.
In this chapter, we present a preliminary campaign results from the sensors stations deployment between Summer 2017 and Spring 2018.

5.2 The spatial context

The Joanina library is located in the heights of the historic center of the University of Coimbra (UC), 120m above the sea level. It was built between 1717 and 1725, and the decoration works lasted for another three years. [75]
It consists of three floors: the Noble floor, the Intermediate floor and the Academic Prison.
The noble floor is the heart of the site and stores circa 40,000 books; it's rich in decorated ceilings and walls, ancient wooden bookshelves with golden details. The Intermediate floor has workplace tables and some cases as a small exhibition [76].

The Academic Prison, worked from 1773 until 1834 [77] and it had a souvenir shop until Summer 2018.
The library is daily open, with 180 visitors each hour maximum: 20 minutes shifts with 60 people, receiving more than half a million tourists in 2017. Visitors are accepted from 9h00 to 13h00 and from 14h00 to 17h30 in Winter and from 9h00 to 19h30 in Summer.
Last average rainfall amount in Coimbra varied between 5.5mm in July 2017 (3 days) and 449mm in March 2018 (26 days). The average outdoor temperature was 26°C in Summer and 10°C in Winter. The average wind speed was 5.1mph in August 2017 and 8.1mph in March 2018. [78]
A peculiarity of this library is the presence of bats inside of it. Despite thinking of them a source of risk for the book collection inside, the bats "are essential for the maintenance of the books since they eat bookworms" but "they leave a thin layer of droppings over everything" [79] that requires daily maintenance and cleaning.

5.3 Monitoring Campaign

Initial environmental conditions studies started at the end of the 20th century and in last years the Rectorate of the University started "a research project focused on the accurate characterization of the indoor environmental conditions within the Baroque library and on the assessment of risk situations, both for heritage and health issues" [76].
In early 2017, a preliminary environmental analysis has been done by the Department of Mechanical Engineering of the University of Coimbra. [76] Subsequently we started a new monitoring campaign led by the Macao Polytechnic Institute in collaboration with the Sorbonne Université, at the beginning of June 2017, placing first three Canarin II in the noble floor, focusing on Particulate Matter (PM) concentrations. In particular, we put one sensor on the mezzanine floor close to the main entrance, one at the bottom of the main entrance and one to the door to the intermediate floor. They are constantly plugged to the library power source even if electricity is available on during opening hours; Rechargeable lithium batteries are installed to extend data availability during the night.
Until November 2017, tourists used to enter from the main big door at the noble floor, then the administration decided to let them enter from the academic prison/souvenir shop to mitigate the impact of the dust from the main courtyard in front of the library.
After November 2017 other six stations have been installed:

- One in the academic prison, close to the new entrance;
- One inside, at the intermediate floor, on a table;

- One outside, at the intermediate floor, laced to a window grid;
- Three more in the noble floor, covering all the area: two on the floor and one on the mezzanine wooden floor.

Collecting data in in the whole area of the library makes it possible to identify which different factors play a role in the PM trend into the different floors. Along with air pollutants, other aiding parameters are gathered and presented in table 5.1

Measure	**Sensor type**	**Unit of measure**	**Monitoring interval**
Temperature	Thermistor	°C	Every 60 sec
Humidity	Thermistor	%	Every 60 sec
Air Pressure	Piezoresistive sensor	hPa	Every 60 sec
$PM_{1.0}$	Light scattering optical analyzer	$\mu g/m^3$	Average value from 30 samples achieved in 30 seconds
$PM_{2.5}$	Light scattering optical analyzer	$\mu g/m^3$	Average value from 30 samples achieved in 30 seconds
PM_{10}	Light scattering optical analyzer	$\mu g/m^3$	Average value from 30 samples achieved in 30 seconds

Table 5.1: Measurements details and monitoring intervals

As future work some FLIR Lepton 3.5 thermal cameras will be installed inside the sensor stations to:

- Check the relationship between heat sources and pollution
- Track the bats behavior inside the library

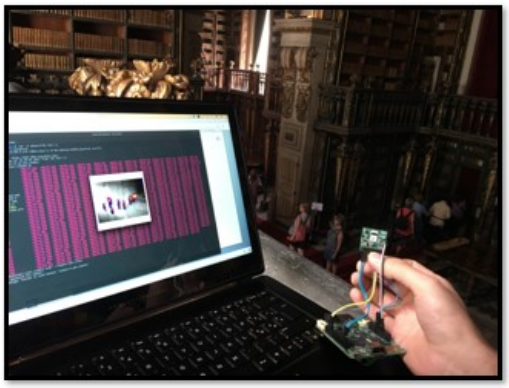

Fig. 5.1: A thermal camera demo inside the library

5.4 Architecture

Canarin II is a powerful low-cost, low-power Arduino/Linux single board running an ARM® Cortex-A9 microprocessor and a Cortex-M4 I/O real-time co-processor. Its design, code and architecture are the joint result between Sorbonne Université, the Macao Polytechnic Institute (MPI) and the Asian Institute of Technology (AIT - Thailand). It detects PM_{10}, $PM_{1.0}$ and $PM_{2.5}$, air pressure, temperature and relative humidity, in an indoor environment.

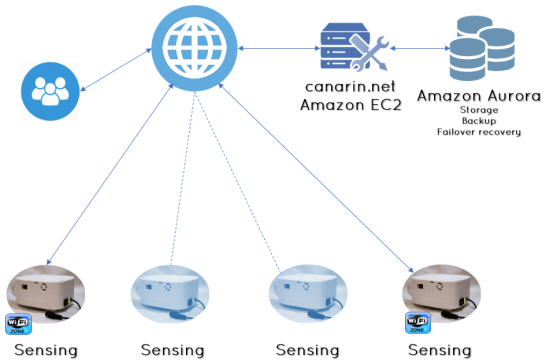

Fig. 5.2: Canarin II architecture

From its previous version [14], Canarin II has been enhanced with a new firmware and new sensors. In particular it has:

- More storage capability: all the systems are equipped with 32GB SD cards;
- WiFi now supports WPA2-Enterprise and 802.1X Simplified
- PM sensor software is improved gaining refined values;
- The Linux kernel and its packages are updated and maintained

Canarin II's core runs an Arduino-like sketch on the real-time co-processor that coordinates every sensors mentioned before. The hardware is based on in-house printed circuit board (PCB) that encapsulates the board, the wiring and the sensors' sockets. Around it, a 3D printed polylactic acid (PLA) box has been designed to tangle the PCB and the 6800mAh lithium battery together. Its size is 19x15x7 cm and it weighs about 900g.
The communication relies on *Eduroam* Wi-Fi offered by the University of Coimbra nearby, even if the board also supports GSM network using the EAP-SIM (EAP Subscriber Identity Module) authentication framework by means of a USB SIM reader; in the current setting *Eduroam* has been preferred.
The communication protocol is based on a customized UDP; every Canarin II interacts with a server that runs on a Amazon EC2 instance, which manages the packets

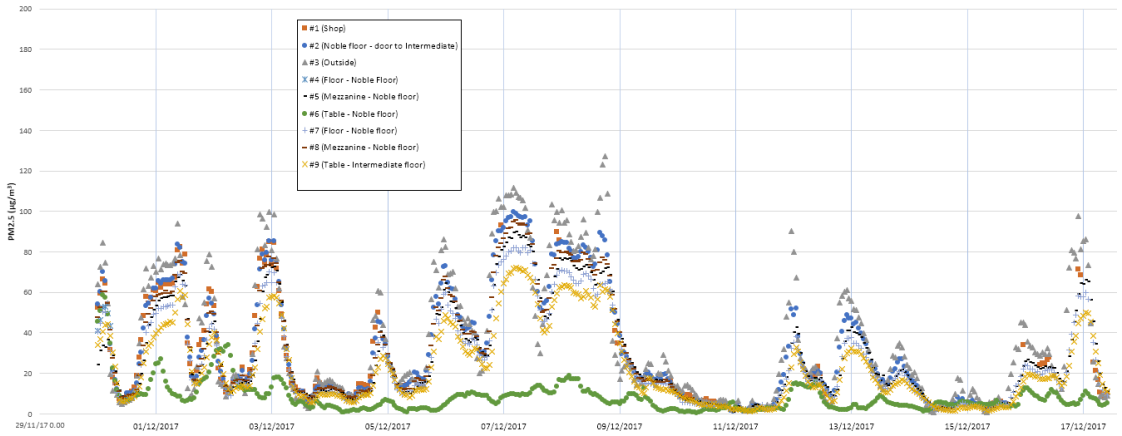

Fig. 5.3: All Canarins $PM_{2.5}$ concentrations values from 30 November 2017 to 18 December 2017

and stores the sensors values into a MySQL database (Amazon Aurora). Amazon Aurora ensures fail-over recovery and DB daily backups.

During the closing hours, since the library doesn't provide power supply, the lithium battery inside every canarin let the sensors sample during the night. In the morning the power has restored and all the data, stored into the SD card, are uploaded to the server.

5.5 Preliminary data analysis

This section presents some preliminary results obtained during the campaign started on Summer 2017 (still active).
Particulate matter - and others parameters - are monitored in four different scenarios:

- In the noble floor - upstairs, on the mezzanine wooden floor by Canarins #2, #4, #6, #7, #8
- In the noble floor - downstairs by Canarins #5, #8
- In the intermediate floor/prisons by Canarins #9, #1
- Outside by Canarin #3

Up to August 2018, we collected 5,574,959 samples between PM_{10}, $PM_{1.0}$ and $PM_{2.5}$. To date (November 2020) the total samples are 10,512,054.
As shown in table 5.2, in the noble floor we detected good air pollution level in 56.42% of the time, according to US-EPA 2016 standard [80] thats defines air quality level (AIQ) into six categories:

Range $PM_{2.5}$	Air Pollution Level	%
0 - 12.0	Good	56,42
12.1 - 35.4	Moderate	36,54
35.5 - 55.4	Unhealthy for sensitive groups	4,65
55.5 - 150.4	Unhealthy	2,30
150.5 - 250.4	Very unhealthy	0,06
250.5 - 500.4	Hazardous	0,03

Table 5.2: Air quality level in the Noble floor from Nov 2017 to Jul 2018

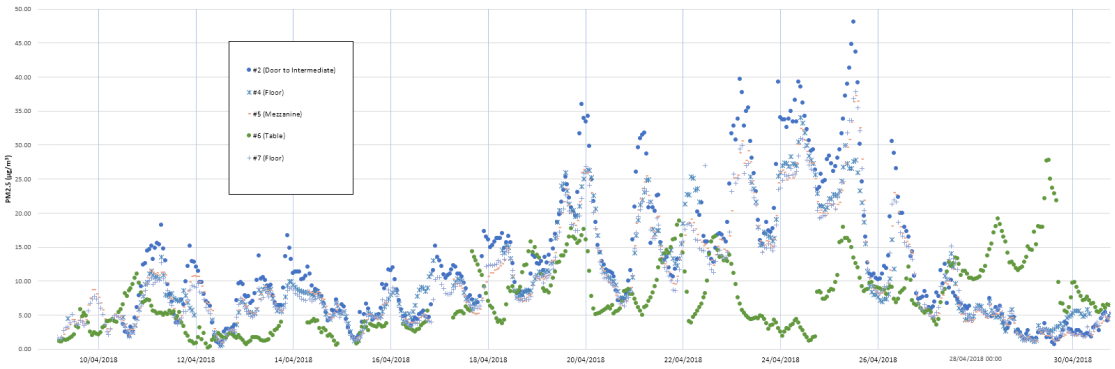

Fig. 5.4: Noble floor Canarins $PM_{2.5}$ values from 9 April 2018 to 30 April 2018

- Good: $PM_{2.5}$ [0 - 12.0]
- Moderate: $PM_{2.5}$ [12.1 - 35.4]
- Unhealthy for sensitive groups: $PM_{2.5}$ [35.5 - 55.4]
- Unhealthy: $PM_{2.5}$ [55.5 - 150.4]
- Very unhealthy: $PM_{2.5}$ [150.5 - 250.4]
- Hazardous: $PM_{2.5}$ [250.5 - 500.4]

Fig. 5.5: Canarin #5 on the mezzanine floor

Figure 5.3 presents the $PM_{2.5}$ concentration trending in the time window between 30 November 2017 and 18 December 2017. At the first glimpse, the reader could see how all the sensors tend to follow the same trend even if not all the Canarins are in the same spot.
High values peaks are present during the night, after the closing time.
The Canarin outside the window (#3 - grey triangle) detects the highest values and it defines the trend maximum boundaries while the Canarin on the table in the Noble floor has the lowest values. The latter does not seem to 'follow the trend', probably due to a malfunction PM sensor. In our past experience, we checked that some sensor in a batch could lead to underestimated values.

In figure 5.4, on the other hand, it shows the behavior focused on the noble floor from 9 April 2018 to 30 April 2018: the sensor close to the passage to the intermediate floor shows the highest values; in fact this is a spot in front of the tourist tour, inside the library. In this time window, the sensor on the mezzanine matches the sensors on the floor: the different height does not affect much the values. Canarin #8 was off on April.
From the two trends it can be state that the concentrations of particles on December are much higher than the ones in April. Therefore, it remains unclear whether this is related to the house heating in cooler months as December that may influence the air pollution in Coimbra; consequently much particulate matter might enter into the library with tourist tours.

5.6 2020 Updates

Fig. 5.6: New Canarin II version 2 on the noble floor

Further data analysis is continuing to date, with other five new Canarin II version 2 installed between the various floors. One extra platform was placed inside the historical Science Museum of the University of Coimbra. Latest Canarin II units are provided with a CO_2 sensor.
Following the users' feedback, we enhanced the canarin II version 1 with extra leds to notify the WiFi status (on/off), the battery level (high/low), and a sampling led warning.
The touristic path inside the library has been rescheduled annually, according to the data processed with our samples; this helps to keep the PM values below the "hazardous" level.

5.7 Final remarks

This chapter proposes an indoor pollution study for the UNESCO heritage Biblioteca Joanina in Coimbra, Portugal. The study was performed using a low-cost edge-based sensing platform and enabled the library managers to understand how to modify the paths of tourists in order to reduce the PM in the library. We also noticed a higher than expected impact of the outside pollution into the library likely due to the ancient doors and window that do not represent a bearer for the $PM_{2.5}$ or PM_{10}. In the next step, we plan to develop a privacy-aware tracking technology to enable the correlation between PM values and the actual paths taken by the tourists. This will require building distributed computing systems to perform the necessary machine learning tasks in a distributed fashion.

Chapter 6

On Assessing the Accuracy of Air Pollution Models Exploiting a Strategic Sensors Deployment

This chapter, likewise the previous one, presents a preliminary experiment done to identify potential problems and issues in setting up a testbed for air pollution measurement and modeling. Our final testbed, part of a joint research activity between the University of Bologna and the Macao Polytechnic Institute, will be composed of three lines of the air pollution sensors Canarin II and it will be used to produce spatio-temporal open data to test third-party air pollution models. Here, we present a preliminary experiment based on a single line of sensors, showing interesting insights into the actual open challenge of air pollution modeling techniques validation, taking into account the effects of air pollutant emissions sources, meteorology, atmospheric concentrations and urban vegetation.

6.1 Introduction

Air pollution is a phenomenon by which solid and liquid particles and gases contaminate the environment with negative effects on population health [81, 67]. The effects can be really serious, even lethal, as stated by the World Health Organization (WHO) that renominated this phenomenon *the invisible killer*, estimating in 7 million the number of deaths every year caused by the exposure to fine particles in polluted air [70]. In the same report WHO also claimed that 9 out of 10 people worldwide breathe polluted air and more than 80% of people live in urban areas where the air quality levels exceed the WHO guideline level [82]. Different strategies can be employed to tackle this problem and achieving sustainable development, such as sustainable transport, more efficient and renewable energy production and use and waste management [83]. The first action that local governments and policymakers should tackle is the gathering of air quality data to reflect their commitment to

air pollution assessment and monitoring [84]. Having the data, it becomes relevant to develop models able to understand and predict the way pollutants behave in the atmosphere, so as to state the actual air quality in an area [85], and then equipping citizens with tailored services (i.e., mobile apps computing personalized pedestrian and cycling paths in the urban environment [86, 71], keeping the citizens in mind, as well as their preferences and needs [87, 88]).

Several mathematics theories and numerical tools have been studied in the literature, under the umbrella term *air pollution modeling*, to understand the causal relationship between emissions, meteorology, atmospheric concentrations, deposition, and other factors [89]. The techniques used are several (see, for example, [90, 91]) but the common goal is to make an assessment of pollutant impact over a given area using a defined set of data.

In the last years, new technologies are emerging, which provide an alternative and cost-efficient approach to measure air pollutants [40]. Being low-cost and adequately precise, this new generation of pollution sensors are completely changing the possibility to be aware of the air quality in the urban environment. In this context, we are involved in the development and test of Canarin II, the result from the collaboration among the University of Bologna, the Universitè Pierre et Marie Curie, the Macao Polytechnic Institute, and the Asian Institute of Technology. The Canarin II architecture works on a UDOO Neo Full, an Arduino-powered Android/Linux single board. It measures PM_1, $PM_{2.5}$, and PM_{10}, pressure, temperature, humidity, and UV (Chapter 3).

The availability of Canarin II sensors drove us in designing and setting up an air quality testbed, with three goals in mind:

(i) To produce a set of spatio-temporal open data in different weather condition in order to provide third parties with air quality data to test their own models. Other measures (e.g. sensed wind, detailed map of the area, vegetation distribution) will be provided in order to offer a complete set of data to appropriately test pollution diffusion models.

(ii) To develop models to measure and test the accuracy and efficacy of air pollution modeling techniques, using outdoor air quality data collected under the presence of specific circumstances that can affect the outcome, such as present and future barriers. To make the defined models even stronger, in the future, such dataset could be integrated with data recorded using mobile sensors provided to users and gathered while moving in the surrounding area (e.g. using sensors on shared bikes (Chapter 3).

(iii) To determine the best configuration needed in term of the number of sensors and distance between each sensor in order to collect air quality data and assess with a high accuracy the validity of air pollution models.

To address the three research issues, we designed a preliminary experiment using a set of sensors that are being deployed around a new building, 30.000 m^2 wide, which

is the new seat of the Campus of Cesena of the University of Bologna (in the Cesena city). The building is located in a particular area, surrounded both by pollutant sources (e.g. a highway and a railway) and by residential/green areas. Due to the peculiar characteristics of the area and the shape of the building, a strategic sensors deployment will let us collect a rich dataset that incorporates phenomena affecting the air quality assessment.

6.2 Air pollution models assessment

As briefly mentioned, several air pollution modeling techniques have been studied and developed to address three main concerns: to assess the existing air quality situation and calculate the population exposure to pollution; to forecast changes in pollution levels and prevent or inform about oncoming predicted critical episodes; to define an air quality planning program. Often these models have been used i) in various forms, ii) with not-standardized and not accurate enough datasets, and ii) with differing and often incomparable quality assurance methods, at both national and local levels. This scenario let emerge the urgent need to harmonize the way these models are validated so as to achieve reliable results.

Different factors affect the outcome resulting from the application of air pollution modeling techniques and need to be considered in the design of the models to avoid uncertain results [92]. Such issues include:

- urban vegetation: vegetation can, directly and indirectly, affect local and regional air quality by altering the urban atmospheric environment [93];
- background concentration: this factor indicates the concentration that would be measured if local sources were not present [94];
- urban layout: buildings can alter the concentration and deposition values [91];
- terrain: the conformation of the area needs to be considerate to simulate the movement of pollutants in the atmosphere [91];
- water source: water, in form of rivers, lakes, or oceans, may transport pollution for long distance, and, sometimes, in high concentrations [95];
- meteorological data: information about wind speed and direction, temperature, humidity are relevant in air pollution modeling [96].

Acquiring precise and variegated enough input data to test the models is a hard task. Some projects have been created with the aim of creating an open data collection of air quality data. One example is represented by OpenAQ[1] with a dataset aggregating air quality measurements, obtained by government agencies,

[1] https://openaq.org/

from 8,589 locations in 67 countries. Even than this project is really interesting and has hight potential as a tool to inform people about monitored pollution, the dataset can not be used for air pollution modeling since it doesn't include important information related to the context the data are collected, neither the accuracy of the used sensors. Unfortunately, this problem is common to several government open data air quality measures collections. For this reason, with this experiment we want to fill the gap providing an open dataset of spatio-temporal data, providing all the information needed to assess in a rigorous way the accuracy of thirty-party air pollution models.

6.3 The spatial context

Our testbed is being set up in the Campus of Cesena of the University of Bologna. It is a new building (partially under construction), located at the border between a park (see A in Figure 6.1, west direction), a residential area (B, sud), a not operating industrial zone (C, east) and the railroad/highway area (D, north). The park is part of the Savio River Reserve, which extends along the River Savio in the part where the reserve enters the urban area of Cesena[2]. The not-operating industrial area is occupied by two buildings originally devoted to fruit storage and distribution. The highway and the railroad enter the city of Cesena going parallel. The highway goes underground to cross the city center nearby the campus building (300 m). The railroad reaches the station 1,2 Km after the campus building. In order to protect the building from pollution coming from the railroad and the highway, an artificial hill (E, north) was created in the north garden, between the building itself and the railroad/highway area.

Figure 6.2 shows a more detailed map, where the main sources of pollution are depicted respectively in white (H, the highway) and yellow (R, the railroad). A minor source of pollution is via Macchiavelli (M, in blue in figure 1). All other streets around the building (depicted in light blue) can be considered irrelevant due to the very limited traffic flow.

The artificial hill E partially protects the building from pollution coming from the D area. Figure 6.3 shows the view of the railroad and the highway taken from the internal part of the courtyard garden. While the hill protects the right part of the view, on the left both the railway and the highway are visible. Similar pictures can be taken from the east side of the building.

6.4 The sensors station

The Canarin II is the result of the collaboration among the Macao Polytechnic Institute, the Asian Institute of Technology, and the Pierre and Marie Curie Sorbonne

[2]https://en.wikipedia.org/wiki/Savio_River_Reserve

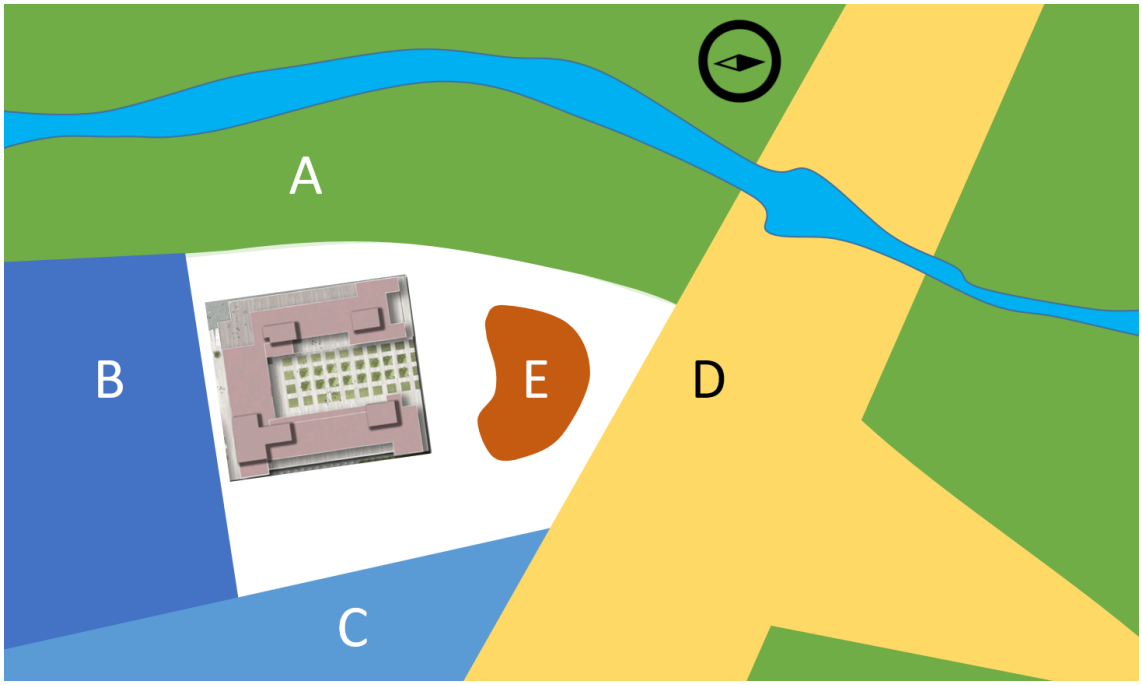

Fig. 6.1: Areas around the campus building.

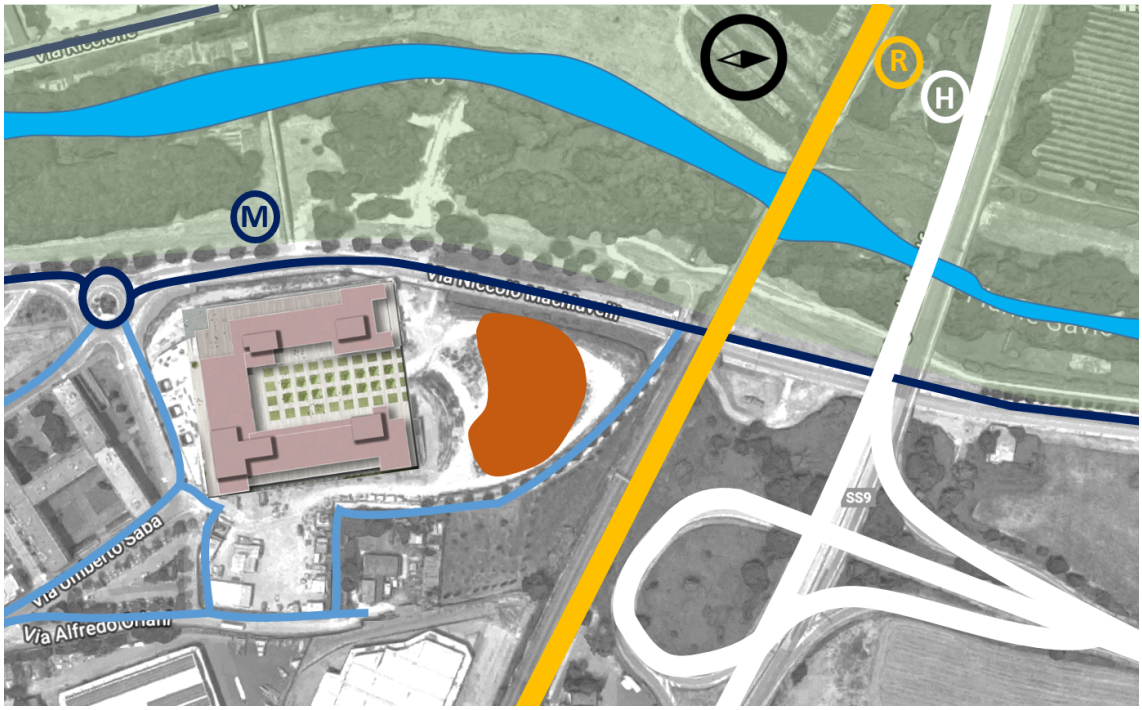

Fig. 6.2: Detail of roads around the building.

Fig. 6.3: The artificial hill created to protect the building from pollution coming from the railroad and the highway.

University.

The sensors station senses different air quality and environmental conditions, such as PM 1.0 (<10 microm) particles, $PM_{2.5}$ (<2.5 microm) particles, and PM_{10} (<10 microm) particles, temperature, relative humidity, and air pressure.

The architecture is based on UDOO Neo Full[3], an Arduino-powered Android/Linux single board computer that overcomes some limitations emerged in the first version of the sensors station [14], such as:

- a new 1GHz ARM Cortex-A9 microprocessor;
- the inclusion of $PM_{1.0}$ (<1.0 microm) particles concentration and formaldehyde sensors;
- a more powerful SD card;
- a new Wi-Fi module for a stronger connection stability;
- a better management of the sensed data: if the Internet connection is missing, the data are saved on the local SD and send to the server when possible.

The new system architecture is, therefore, structured around three layers: (i) all the sensors run on an Arduino UNO-compatible platform that clocks at 200 MHz, based on a Cortex-M4 I/O real-time co-processor; (ii) a Linux based OS stores the data into files and it establishes a connection to the server in order to send data; (iii) a server side that provides a data repository and data intelligence as well. The sensors station design is based on an in-house printed circuit board (PCB) that hosts the board and the sensors welded. The communication is based on Wi-Fi and the board is also configured to be connected to GSM network using the EAP-SIM

[3]https://www.udoo.org/udoo-neo/

(EAP Subscriber Identity Module) authentication framework by means of a USB SIM reader. The communication protocol is based on a customized UDP and it communicates to an enhanced MySQL database version.

We also designed an in-house 3D printed PLA (polylactic acid) box to wrap the battery, the PCB, and the sensors together. Everything fits in a 19x15x7 cm and it weighs about 900g due to the battery and the enclosure; a smaller and more portable 3D printed box is about to be produced.

6.5 The sensors deployment

As shown in Figure 6.4, our testbed will be set up using three outdoor spaces: the west terrace, facing via Macchiavelli (1.WT - located at second floor of the campus building), the courtyard garden (2.CG - located at the ground floor) and the east terrace (3.ET - located at second floor of the campus building). Canarin II pollution sensors station will be located in three parallel lines of 4, one every 20 meters. The testbed is completed by an anemometer, to measure the speed of the wind, located on the rooftop of the building (4.AN, located at the roof of the third floor of the campus building).

The setup of the whole testbed will be completed in the next months, accordingly with the completion of the building construction. This chapter presents some preliminary tests conducted on a partial sensors' infrastructure, located in the WT (this area of the building has been already completed). The dislocation of one line of sensors stations in the WT area is shown in Figure 6.5, with a distance of 20 meters between each Canarin II. Having four sensors station in line allow as to overcome issues related to the accuracy and calibration of the specific sensor. We estimated in 20 meters the right distance to collect data affected by different factors. In the next future, more tests will be made to define the best distance needed between each sensor, and if this value is affected by the position of the sensors in the different areas (i.e., WT, GT and ET), settled to capture different urban scenarios and pollution sources.

In a month of data gathering, we already gathered 714,240 complex entries in our database. Each entry includes several values (such as PM_1, $PM_{2.5}$, PM_{10}, humidity, temperature, wind direction and speed, air pressure) contextualized in space (the GPS coordinates) and time.

6.6 Conclusion and future works

In this chapter, we presented a preliminary experiment to strategically deploy sensors stations in a new university campus. At this stage, the campus is composed by a single building that is located in a peculiar area, allowing to sense air quality data affected by different factors, including vegetation, different mobile pollution sources,

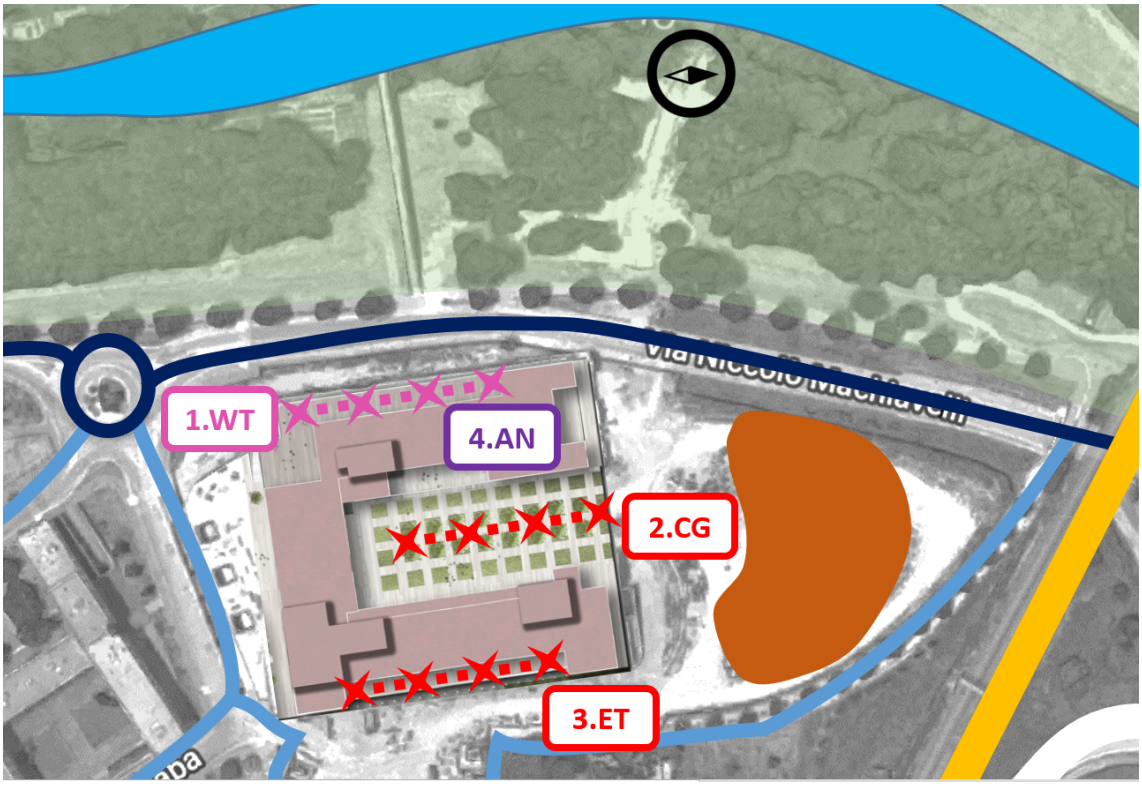

Fig. 6.4: Testbed design and experimental setup.

Fig. 6.5: The deployment of one line of sensors

a specific urban layout, and the proximity to a river natural reserve.

With this research project, we intend to address three open issues: i) producing an accurate open data collection that can be exploited by third parties to assess their own air pollution models; ii) developing models to measure the accuracy and test the efficiency of air pollution modeling techniques; iii) defining strategic deployment configurations of air quality and environmental sensors (e.g., the distance between each sensor, the number of sensors, positions).

As an initial deployed, we used a line composed by four Canarin II sensors stations. Each sensors station is able to accurately sense PM_1, $PM_{2.5}$, PM_{10}, relative humidity, temperature. Moreover, we augmented the sensing platform with an anemometer, to measure the speed of the wind. In a month, we already collected 714,240 spatio-temporal entries, where each entry is composed of all the sensed environmental conditions at a given time and location.

We are planning to extend the actual deployment including other two lines of sensors located in strategic positions in the building, facing areas with different characteristic (such as vegetation, layout, exposure to pollutant sources). Moreover, we are considering to integrate our dataset with data gathered using mobile air quality sensors to create an even more accurate and variegated collection of open data.

Chapter 7

Over The Air EV monitoring and analytics: the Nissan Leaf case

The controller Area Network (CAN) is the most widely used communication standard within automobiles and light trucks. It allows devices and microcontrollers to communicate with each other's applications without a host computer. In particular, it synchronizes every Engine Control Units (ECU), which are the hardware computing platforms that control every subsystem of the car electronics.

In this context we outline the second of the two key sustainability C-Continuum macro-scenarios: thanks to the doable combination between a powerful, easy to handle, CAN-ready platform, a fleet of electric vehicles offered by the University of Bologna, and a cheap 4G/LTE data plan, we started a fine-time granularity battery data collection campaign. In this case, the mobile nodes are some real Nissan Leaf in which we installed some diagnostic IoT devices to gather the lithium battery's data along, with many other useful parameters.

7.1 The CAN Protocol

The CAN protocol was release in 1986 during the Society of Automotive Engineers (SAE) conference in Detroit, Michigan after three years of development by Robert Bosch GmbH[97]. It was designed for multiplex electrical wiring within automobiles and Mercedes-Benz W140 was the first production vehicle to feature CAN-bus.
At the moment CAN 2.0[98] is the latest version available and its specification includes two different identifiers: CAN 2.0A, which has an 11-bit identifier while CAN 2.0B describes the 29-bit identifier. In 1993, the International Organization for Standardization (ISO) released the CAN standard ISO 11898 which covers three layers: the data link layer (ISO 11898-1[99]), the physical layer for high-speed (up to 1MBit/s) CAN (ISO 11898-2[100]), and for low-speed (up to 125 kBit/s) fault-tolerant CAN (ISO 11898-3[101]). Higher layers not standardized by CAN and they are covered by additional standards and conventions, e.g. CANopen[102].

The CAN protocol connects every device on a shared bus with a maximum data rate of 1 Mbit/s; common data rates are also 500Kbit/s and 125kBit/s for a max 500m long wire. The frame transmission has a broadcasting collision-free priority-based approach: if more than one device transmits at the same time the highest priority device is able to send the message while the others wait for the next sending interval. Frames are received by all devices, including by the transmitting device.
The CAN data format is summarized in Figure 7.1:

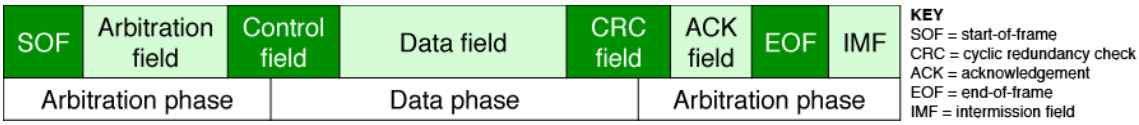

Fig. 7.1: The structure of CAN data frames[103]

- Frame begins with a start bit;
- Message identifier 11 Bit (CAN 2.0A), or 29 Bit (CAN 2.0B)
- Control bits, e.g. the message type (Request, Data, Error, Overload), the message length, the FDF (FD format) bit distinguishing the two data link layer protocols, Classical CAN and CAN FD, etc;
- The payload is generally 8 byte per message
- The CRC for error detection done by the receiver. All nodes check the protocol conformance of messages and bit stuffing. If any node detects some errors, it transmits the error signal over the bus. All the nodes that detect the error signal, discard the message.

The CAN data link layer provides multi-master capability: any node can access the bus at any time if it is idle. If several nodes try to communicate at the same moment, the highest priority message wins the bus arbitration and starts transmitting. The message priority is assigned by the system designer, selecting a CAN identifier (CAN-ID). The lower the number of CAN-ID, the higher the priority. The value of "0" is the highest priority.

7.1.1 OBD-II: on-board diagnostic

Nowadays, the CAN protocol is the most used in the on-board diagnostics, namely OBD. It supplanted older protocols as J1850 PWM, J1850 VPW[104], ISO9141-2[105], ISO14230-4[106] (also known as Keyword Protocol 2000). The OBD version II is now the mandatory standard embedded for all cars and trucks; starting from the USA in 1996, it's now included into European cars from 2001 and Asian ones from 2011.

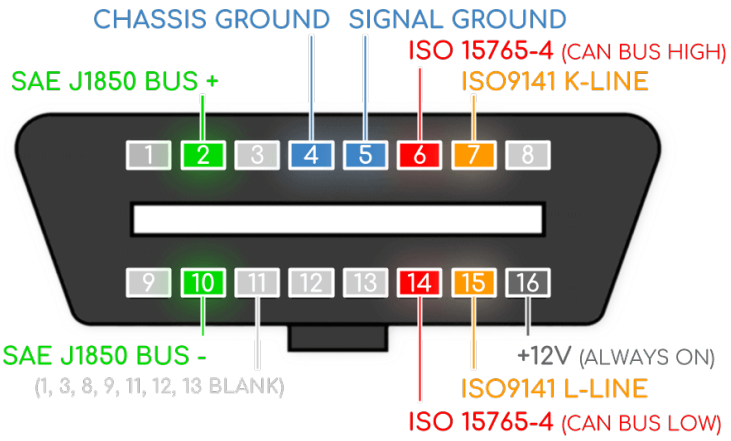

Fig. 7.2: OBD2 pinout and supported protocols[107]

The OBD-II is used for a variety of needs like Diagnostic Trouble Codes (DTC) reporting; the car repair technician reads the DTCs to understand the origin of the issue when a warning led turns on into the dashboard; the OBD-II protocol is also used for CAN messages spoofing[108] or to tweak the car[109], to disable alerts or to change car's behavior in some cases.
Recently, the IoT market released lots of OBD-II Bluetooth and WiFi dongles to access the ECUs information inside the vehicle. Thanks to some mobile applications developed for every car model, the driver can access the state of health information for various vehicle sub-systems.

7.2 Energy consumption compaign

At the University of Bologna, we started an electric vehicle monitoring campaign at the beginning of 2020. The goal of the campaign, still ongoing, is to retrieve different EV usage data and evaluate the correlation between the driving style and power consumption. Along with the engine power, we also collected the lithium battery data from the Battery Management System and the HVAC power.
As a sensing platform, we used the iWave OBD-II[110], manufactured in India and Japan. The iWave's OBD-II is a little device with an ARM Cortex-A7 processor embedded, it runs a light Yocto Poky Linux distribution with *busybox*[1].
It has a 4G/LTE CAT4/CAT1 sim modem, a GPS receiver and it supports BLE 4.2. It has also an OBD2 interface that we requested to connect to the Leaf. It is integrated with sensors onboard that enable the user to track and monitor the driving behavior and car diagnostics remotely, along with an accelerometer, gyroscope, and a magnetometer. The hardware setup was easy; since the busybox version installed was very lightweight, we had to cross-compile lots of libraries, and

[1] `https://busybox.net/`

even python3.

The University of Bologna has at its disposal a fleet of 30 Nissan Leaf 2018.
The Nissan Leaf[2] - where "LEAF" is the acronym for *Leading, Environmentally friendly, Affordable, Family car* - is a five-door hatchback electric car produced by Nissan and initially introduced in Japan and the United States in December 2010. Nissan electric vehicles have existed since 1946 and the Leaf was preceded by the Altra and the Hypermini. Two Leaf generations were produced in the last decade: the first one from 2010 to 2017 and the current one started with our model, starting from 2018.
Moreover, two different 2018 models are depending on the battery pack capacity: the former has 40kWh while the latter has 62kWh. Our cars belong to the first type and it lacks of the battery pack #3.
Some other technical details are summed up in the following table:

Measure	**Unit**	**Value**
Type of motor	-	IPM (EM57)
Maximum power	kW(CV)/rpm	110 (150)/3283-9795
Maximum Torque	Nm/rpm	320/0-3283
CO2 emission	g/km	0
Battery capacity	kWh	40
Recharge time	h	16 (240V and 3kW)
Maximum speed	km/h	144
Maximum acceleration 0-100km/h	s	7.9
Average consumption	km/kWh	9.5
Autonomy	km	378
Mass	kg	1535

Table 7.1: Nissan Leaf 2018 tech specs

On the diagnostic-side, every Nissan Leaf has been deeply studied by many *hackers*, who wanted to take advantage of every detail from the car owned, starting from the very first models. So far, Nissan has not openly released any CAN message semantic protocol but the "Service Manual". The service manual LAN section has thorough information regarding the connectivity, pinouts, and functional blocks on the CANbus. However, it does not describe the actual message packets.
Over time, a large community has been created in online forums and blogs, where every Nissan driver can share his driving experience, and the OBD monitored data usage, along with the battery consumption.

[2] `https://www.nissanusa.com/vehicles/electric-cars/leaf.html`

Fig. 7.3: iWave OBD2

The CAN network is accessible from the 16-pin DLC (Data Link Connector), labeled (M4) in the service manuals, that is the OBD connector aforementioned. This connector is located above the driver's left knee. In detail, the Nissan Leaf has three CAN buses that connect all the major modules and relay information between them:

- Primary CAN Bus is on the standardized (SAE J2284[111]) connector pins, CAN-L (14) and CAN-H (6); this bus carries traffic between the VCM, DLC, BCM, ABS, Steering, and Dash/Nav. It has most of the interior operations, displays, and controls.
- EV CAN (on pins EV CAN-L (12), EV CAN-H (13)); this bus has communication including the VCM, HVBAT, OBC, and TCU. To connect to this bus, it needs to make a custom J1962 cable, as it is not the 'default' CAN bus for diagnostic tools.
- AV CAN (on pins AV-CAN-H (11) and AV-CAN-L (3)); that reports information between the Audio-Video Navigation unit and the "multifunction switch".

Pin 16 has constant +12V DC power from the battery. It is protected by a 10A fuse. Pin 4 is Chassis Ground, and Pin 5 is Signal Ground. Both of the CANbus networks operate at 500kbps.
Even if we were expecting some broadcasted message once connected to the primary CAN bus, we found no passive vitals on it, neither with an oscilloscope nor reading the CAN interface with our OBD-platform. This happens because Nissan introduced a central gateway in 2018+ models that communicate with the multiple buses to access the various vehicle ECUs. Hence, they replaced the passive message approach for active polling. This required a huge effort to retrieve all the variables described below since we could not rely on the spreadsheet created in the last ten years from passive spoofing. Instead, we used a custom OBD CAN reader to query

each arbitrary ID, looking for the correct variables.
Before going further with the variables and the corresponding messages, let's remember that a CAN message is composed of an 11-byte identifier and a 8-byte data field. There is no notion of message addresses in CAN; the message is said to be "contents-addressed".
The LEAF uses the 11-bit CAN frame exclusively, so the identifiers could range from 0 to 0x7FF; not all the identifiers can be queried, i.e. they neither reply with an error message nor any message at all. Every ECU has a defined ID that identifies it and it replies with a different, but consistent id. Here in details:

ECU	**ID Query**	**ID Response**
Vehicle Control Module (VCM)	0x797	0x79A
Body Control Module (BCM)	0x743	0x763
AntiBlockierSystem (ABS)	0x740	0x760
Li-ion Battery Controller (LBC)	0x79B	0x7BB
Traction Motor Inverter (INV/MC)	0x784	0x78C
Meter	0x745	0x765
HVAC	0x744	0x764

Table 7.2: Nissan Leaf 2018 ECU IDs

If the query is not well-formatted, the ECU will reply with:

0x7XX 03 7F 21 11 FF FF FF FF

In the payload, the unused bytes are usually 0xFF.
In order to comply with the Nissan protocol, the message query must also follow these rules:

- The first byte corresponds to the length of the payload (usually 02 or 03);
- The second byte is usually 0x21 for a long response or 0x22 for a single message response;
- The third and fourth bytes are the variable ID that we reverse-engineered

For example, the correct query to ask for the gear is:

0x797 03 22 11 56 FF FF FF FF

The VCM replies with no errors:

0x79A 04 62 11 56 01 FF FF FF

where 0x79A is the corresponding VCM's answer message ID, 04 is the payload byte length, 62 corresponds to 22+40 (it is always the third question byte plus 40), 11 and 56 are the gear id while 01 is the variable we were looking for. In details: 1=Park, 2=Reverse, 3=Neutral, 4=Drive.
In some cases, the ECUs reply with many messages to a single request; if it happens, the first byte of the first answer is usually 0x10. In order to request the whole response, if at least 8-byte answer has been received, we need

to wait at least 0.5ms and then we send an extra query formatted as below:
0x7XX 30 00 00 00 00 00 00 00
Multiple readings of the CAN buffer are required to get all the answers.
If we may need to ask one message at a time, we can also query multiple
0x7XX 30 01 00 00 00 00 00 00

7.3 CAN variables from the Nissan Leaf 2018

All the variables we found with the matching query CAN message are listed below; all the bytes involved in the formulas are highlighted in each CAN answers.

7.3.1 Vehicle Control Module (VCM)

Name: Power_Software
Type: Boolean
Description: Power_SW is True if the Leaf's power switch is ON otherwise it is False. The fifth reply byte value is 0x80 if the car is powered on.
Query: 0x797 03 22 13 04 00 00 00 00
Answer: 0x79A 05 62 13 04 **80** FE 00 00
Formula: `Power_SW = ( payload[4] & 128 == 128 );`

Name: Gear
Type: Integer
Description: Gear position (0=not read yet, 1=Park, 2=Reverse, 3=Neutral, 4=Drive, 7=B/Eco). The fifth reply byte value is the gear value. In the example gear is 1.
Query: 0x797 03 22 11 56 00 00 00 00
Answer: 0x79A 04 62 11 56 **01** 00 00 00
Formula: `Gear = payload[4]`

Name: Bat_12V_Voltage (V)
Type: Integer
Description: The voltage of the 12 volt battery as read by the Leaf's Vehicle Control Module ECU. The Bat_12V_Voltage is the fifth reply byte value times 0.08. In the example Bat_12V_Voltage is 13.04V
Query: 0x797 03 22 11 03 00 00 00 00
Answer: 0x79A 04 62 11 03 **A3** 00 00 00
Formula: `Bat_12V_Voltage = ( payload[4] * 0.08 )`

Name: Bat_12V_Current (A)
Type: Integer
Description: 12 volt battery current. Minus is drain and positive is charging. In

the example Bat_12V_Current is -0.5546875A
Query: 0x797 03 22 11 83 00 00 00 00
Answer: 0x79A 05 62 11 83 **FF 72** 00 00
Formula:

```
Bat_12V_Current_temp = ( payload[4] << 8 ) | payload[5];
if (Bat_12V_Current_temp & 32768 == 32768):
    Bat_12V_Current_temp = Bat_12V_Current_temp | -65536; //Negative value
Bat_12V_Current = Bat_12V_Current_temp / 256
```

Name: Quick_Charge
Type: Integer
Description: Number of quick charges (QC). Each time a quick charge occurs this number increments by 1, even not fully. In the example Quick_Charge is 1.
Query: 0x797 03 22 12 03 00 00 00 00
Answer: 0x79A 05 62 12 03 **00 01** 00 00
Formula: `Quick_Charge = (payload[4] << 8) | payload[5]`

Name: L1_L2_Charge
Type: Integer
Description: Number of L1/L2 connections and charges, even not fully. In the example L1_L2_Charge is 115.
Query: 0x797 03 22 12 05 00 00 00 00
Answer: 0x79A 05 62 12 05 **00 73** 00 00
Formula: `L1_L2_Count = (payload[4] << 8) | payload[5]`

Name: Ambient_C_Temp (°C)
Type: Integer
Description: Ambient temperature comes from the Leaf's outside temperature sensor and is firstly got in °F. It's then converted to °C. In the example the Ambient_C_Temp is 24 °C.
Query: 0x797 03 22 11 5D 00 00 00 00
Answer: 0x79A 04 62 11 5D **81** 00 00 00
Formula:

```
Ambient_F_Temp = (payload[4] * 0.9) - 40.9;
Ambient_C_Temp = (Ambient_F_Temp - 32) * 5/9
```

Name: EstPwr_AC_50W (W)
Type: Integer
Description: Estimated Air Conditioning system power (COOL)
Query: 0x797 03 22 12 61 00 00 00 00
Answer: 0x79A 04 62 12 61 **02** 00 00 00
Formula: `EstPwr_AC_50W = payload[4]*50`

Name: EstPwr_Htr_250W (W)
Type: Integer
Description: Estimated Cabin PTC heater power (HEAT)
Query: 0x797 03 22 12 62 00 00 00 00
Answer: 0x79A 04 62 12 62 **01** 00 00 00
Formula: `EstPwr_Htr_250W = payload[4]*250`

Name: Aux_Pwr_100W (W)
Type: Integer
Description: Power used by the auxiliary equipment (Lights, Radio, Navigation system, rear defroster...)
Query: 0x797 03 22 11 52 00 00 00 00
Answer: 0x79A 04 62 11 52 **02** 00 00 00
Formula: `Aux_Pwr_100W = payload[4]*100`

Name: AC_Pwr_250W (W)
Type: Integer
Description: A/C system Power (250W): Power used by the Air Conditioning System power. This includes the power used by the cabin PTC Heater.
Query: 0x797 03 22 11 51 00 00 00 00
Answer: 0x79A 04 62 11 51 **01** 00 00 00
Formula: `AC_Pwr_250W = payload[4]*250`

Name: AC_Compressor_Air_pressure (MPa)
Type: Integer
Description: A/C Compressor Air Pressure(0.1MPa): A/C Compressor high side pressure in MPa. Multiply by 14.50377 to get PSI.
Query: 0x797 03 22 11 51 00 00 00 00
Answer: 0x79A 04 62 11 51 00 00 **00** 00
Formula: `AC_Compressor_Air_pressure = payload[6]`

Name: PlugState
Type: Integer
Description: Plug state of J1772 (L1/L2) charge port (0=Not plugged, 1=Partial Plugged, 2=Plugged)
Query: 0x797 03 22 12 34 00 00 00 00
Answer: 0x79A 04 62 12 34 **02** 00 00 00
Formula: `PlugState = payload[4]`

Name: Charge_Mode
Type: Integer
Description: Charging mode. (0=Not charging, 1=Level 1 charging [100-120 volts], 2=Level 2 charging [200-240 volts)], 3=Level 3 Quick Charging)

Query: 0x797 03 22 11 4E 00 00 00 00
Answer: 0x79A 04 62 11 4E **02** 00 00 00
Formula: `Charge_mode = payload[4]`

Name: RPM (r/min)
Type: Integer
Description: Motor revolutions per minute
Query: 0x797 03 22 12 55 00 00 00 00
Answer: 0x79A 05 62 12 55 **FF AB** 00 00
Formula:

```
RPM = ( payload[4] << 8 ) | payload[5];
if (RPM & 32768 == 32768):
    RPM = RPM | -65536;
```

Name: VIN
Type: String
Description: The vehicle identification number indicates which vehicle the data came from. The string of HEX must be converted to UTF-8.
Query: 0x797 02 21 81 00 00 00 00 00
0x797 02 30 00 00 00 00 00 00 00
Answer:
0x79A 10 15 61 81 **53 4A 4E 46**
0x79A 21 **41 41 5A 45 31 55 30**
0x79A 22 **30 35 33 39 32 37 00**
0x79A 23 00 00 00 00 00 00 00
Formula: `(534A4E4641415A45315530303533393237).decode('utf-8')`

Name: OBC_Out_Pwr (W)
Type: Integer
Description: Charging power coming into the Leaf from the on-board charger, when the car is charging.
Query: 0x797 03 22 12 36 00 00 00 00
Answer: 0x79A 05 62 12 36 **00 15** 00 00
Formula: `OBC_Out_Pwr = (( payload[4] << 8 ) | payload[5])*100;`

Name: Motor_Pwr (W)
Type: Integer
Description: Driving motor power.
Query: 0x797 03 22 11 46 00 00 00 00
Answer: 0x79A 04 62 11 46 **00 27** 00 00
Formula: `Motor_Pwr = (( payload[4] << 8 ) | payload[5])*40;`

Name: Speed (km/h)

Type: Integer
Description: Car speed
Query: 0x797 03 22 12 1A 00 00 00 00
Answer: 0x79A 05 62 12 1A **00 01** 00 00
Formula: `Speed = (( payload[4] << 8 ) | payload[5])/10;`

Name: AC
Type: Boolean
Description: AC status (On/Off)
Query: 0x797 03 22 11 06 00 00 00 00
Answer: 0x79A 05 62 11 06 **01** 7f 00 00
Formula: `AC = payload[4]`

Name: RearHeater
Type: Boolean
Description: Rear heater status (On/Off)
Query: 0x797 03 22 11 0F 00 00 00 00
Answer: 0x79A 04 62 11 0F **A2** 00 00 00
Formula: `RearHeater = ( HEX(payload[4]) == '0xA2' )`

Name: ECO
Type: Boolean
Description: ECO mode status (On/Off)
Query: 0x797 03 22 13 18 00 00 00 00
Answer: 0x79A 05 62 13 1B **10** 39 00 00
Formula: `ECO = ( HEX(payload[4]) == '0x10' or HEX(payload[4]) == '0x11' )`

Name: e-Pedal
Type: Boolean
Description: e-Pedal mode (On/Off). The e-Pedal[112] allows the driver to start, accelerate, decelerate and stop using only the accelerator pedal.
Query: 0x797 03 22 13 1A 00 00 00 00
Answer: 0x79A 05 62 13 1A **04** 14 00 00
Formula: `e-Pedal = ( HEX(payload[4]) == '0x4' )`

7.3.2 Body Control Module (BCM)

Name: Odometer (km)
Type: Integer
Description: Total Km run by the car from the production.
Query: 0x743 03 22 0E 01 00 00 00 00
Answer: 0x763 06 62 0E 01 **00 04 DB** 00

Formula: `Odometer = (payload[4] << 16 | ((payload[5] << 8) | payload[6]))`

Name: TP_FR (kPa)
Type: Integer
Description: Front right tire pressure
Query: 0x743 03 22 0E 25 00 00 00 00
Answer: 0x763 04 62 0E 25 **92** FF FF FF
Formula: `TP_FR = (payload[4] * 0.068947576 ) * 100 / 4`

Name: TP_FL (kPa)
Type: Integer
Description: Front left tire pressure
Query: 0x743 03 22 0E 26 00 00 00 00
Answer: 0x763 04 62 0E 26 **94** FF FF FF
Formula: `TP_FL = (payload[4] * 0.068947576 ) * 100 / 4`

Name: TP_RR (kPa)
Type: Integer
Description: Rear right tire pressure
Query: 0x743 03 22 0E 27 00 00 00 00
Answer: 0x763 04 62 0E 27 **94** FF FF FF
Formula: `TP_RR = (payload[4] * 0.068947576 ) * 100 / 4`

Name: TP_RL (kPa)
Type: Integer
Description: Rear left tire pressure
Query: 0x743 03 22 0E 28 00 00 00 00
Answer: 0x763 04 62 0E 28 **94** FF FF FF
Formula: `TP_RL = (payload[4] * 0.068947576 ) * 100 / 4`

Name: Range (km)
Type: Integer
Description: Remaining km to turtle mode
Query: 0x743 03 22 0E 24 00 00 00 00
Answer: 0x763 05 62 0E 2E **00 DD** 00 00
Formula: `Range = (( payload[4] << 8 ) | payload[5)/10;`

7.3.3 Li-Ion Battery Controller (LBC)

The Li-ion battery controller accepts the multi-message query only. In fact, the LBC transmits many groups: the first one contains lots of High Voltage battery data as SOC, currents, and voltage; the second replies with all the battery's cells voltages in

millivolt, the third and the fifth one are still unknown, the fourth contains the four battery packs temperatures, and the last one tells which cell has the shunt active. There are also two more groups: group 61, which replies with lots of CAN messages (up to 48); here we found the SOH value, and group 84 that replies with the HV battery production serial.

Group 1

Query:
0x79B 02 21 01 00 00 00 00 00
0x79B 30 00 00 00 00 00 00 00
Answer:
0x7BB 10 35 61 01 FF FF FC 18
0x7BB 21 02 AF FF FF FB 62 FF
0x7BB 22 FF F0 DD 0B 1C 30 D4
0x7BB 23 95 1D 33 06 03 95 00
0x7BB 24 01 70 00 26 9A 00 0C
0x7BB 25 44 B5 00 11 0B B8 80
0x7BB 26 00 01 FF FF FB 62 FF
0x7BB 27 FF FC AA 01 AD FF FF

Name: Hx (%)
Type: Integer
Description: High Voltage Battery's health. In a factory new condition, the battery's HX is 100%. The number is thought to be an indication of battery internal resistance with 100 indicating a new battery with the lowest resistance. As the value decreases the internal resistance is increasing. As the internal resistance increases more energy is wasted as heat inside the battery instead of powering the Leaf.
Formula:
0x7BB 24 01 70 00 **26 9A** 00 0C
`HX = ((payload[4] << 8) | payload[5]) / 102.4`

Name: SOC (%)
Type: Integer
Description: State of Charge (SOC) of the HV Battery
Formula:
0x7BB 24 01 70 00 26 9A 00 **0C**
0x7BB 25 **44 B5** 00 11 0B B8 80
`SOC = (payload_24[7] << 16 | ((payload_25[1] << 8) | payload_25[2]))/10000`

Name: Ahr (Ah)
Type: Integer

Description: Capacity of HV Battery: e.g. how much energy the battery could hold when fully charged.
Formula:
0x7BB 25 44 B5 00 **11 0B B8** 80

```
AHR = (payload[4] << 16 | ((payload[5] << 8) | payload[6]))/10000
```

Name: GIDs
Type: Integer
Description: GIDs is a Nissan's number that indicate the energy in the Leaf battery. Nissan multiplies this value by 80 Wh to get current capacity of the HV Battery. Discovered by Gary Giddings. The Gid measurement is temperature dependent: understates charge in hot weather and overstates it in cold weather. Formula:

```
GIDS = ((4.425*(SOC * 0.01)) * AHR)
```

Name: HV_Bat_Current_1 (A)
Type: Integer
Description: High Voltage battery current; it's positive if the car is running/driving while it's negative when it's regenerating (by braking) or charging.
Formula:
0x7BB 10 35 61 01 **FF FF FC 18**

```
HV_Bat_Current_1 = (payload[4] << 24) | (payload[5] << 16 | ((payload[6] << 8) | payload[7]))
if(HV_Bat_Current_1 & 0x8000000 == 0x8000000):
    HV_Bat_Current_1 = ( HV_Bat_Current_1 | -0x100000000 ) / 1024
else:
    HV_Bat_Current_1 = HV_Bat_Current_1 / 1024
```

Name: HV_Bat_Current_2 (A)
Type: Integer
Description: High Voltage battery current; it's positive if the car is running/driving while it's negative when it's regenerating (by braking) or charging. It's not clear why there are two different HV current values.
0x7BB 21 02 AF **FF FF FB 62** FF

```
HV_Bat_Current_2 = (payload[3] << 24) | (payload[4] << 16 | ((payload[5] << 8) | payload[6]))
if(HV_Bat_Current_2 & 0x8000000 == 0x8000000):
    HV_Bat_Current_2 = ( HV_Bat_Current_2 | -0x100000000 ) / 1024
else:
    HV_Bat_Current_2 = HV_Bat_Current_2 / 1024
```

Name: HV_Bat_Voltage (V)
Type: Integer
Description: High Voltage battery voltage
Formula:
0x7BB 23 **95 1D** 33 06 03 95 00

```
HV_Bat_Voltage = ((payload[1] << 8) | payload[2]) / 100
```

Name: Insulation
Type: Integer
Description: Unknown
Formula:
0x7BB 23 95 1D 33 06 **03 95** 00

```
Insulation = ((payload[5] << 8) | payload[6])
```

Group 2

Name: Cells voltage (mV)
Type: Integer
Description: These are the 96 cell pair voltages measured in millivolts.
Query:
0x79B 02 21 02 00 00 00 00 00
0x79B 30 00 00 00 00 00 00 00
Answer:
0x7BB 10 C6 61 02 **10 6F 10 6E**
0x7BB 21 **10 6D 10 6F 10 6E 10**
0x7BB 22 **6F 10 70 10 6E 10 6F**
...
Formula:

```
CV_array[0] = (( payload_10[4] << 8 ) | payload_10[5] )
CV_array[1] = (( payload_10[6] << 8 ) | payload_10[7] )
CV_array[2] = (( payload_21[1] << 8 ) | payload_21[2] )
CV_array[3] = (( payload_21[3] << 8 ) | payload_21[4] )
CV_array[4] = (( payload_21[5] << 8 ) | payload_21[6] )
CV_array[5] = (( payload_21[7] << 8 ) | payload_22[1] )
```

...

So far, the **group 3**'s data is unknown.

Group 4

Name: Packs temperature (°C)
Type: Integer
Description: HV Battery temperature sensors from the four packs.
Query:
0x79B 02 21 04 00 00 00 00 00
0x79B 30 00 00 00 00 00 00 00
Answer:
0x7BB 10 1F 61 04 **02 0D** 13 **02**

0x7BB 21 **03** 14 **FF FF** FF **02 0B**
0x7BB 22 13 13 00 FF FF FF FF
Formula:

```
Temp_raw_1 = (payload_10[4]) << 8 | (payload_10[5])
Temp_raw_2 = (payload_10[7] << 8 | payload_21[1])
Temp_raw_3 = (payload_21[3] << 8 | payload_21[4])
Temp_raw_4 = (payload_21[6] << 8 | payload_21[7])
Pack_Temp_C_1 = (( Temp_fromRAW_to_F(Temp_raw_1) - 32 ) * 5) / 9
Pack_Temp_C_2 = (( Temp_fromRAW_to_F(Temp_raw_2) - 32 ) * 5) / 9
Pack_Temp_C_3 = (( Temp_fromRAW_to_F(Temp_raw_3) - 32 ) * 5) / 9
Pack_Temp_C_4 = (( Temp_fromRAW_to_F(Temp_raw_4) - 32 ) * 5) / 9
```

The conversion from RAW temperature to °F is reported in the algorithm 4.

Algorithm 4: Temp_fromRAW_to_F(i)

```
    if i == 1021 then
        return 1.0
3:  else if i ≥ 589 then
        return 162.0 - (i * 0.181)
    else if i ≥ 569 then
6:      return 57.2 + ((579 - i) * 0.18)
    else if i ≥ 558 then
        return 60.8 + ((558 - i) * 0.16363636363636364)
9:  else if i ≥ 548 then
        return 62.6 + ((548 - i) * 0.18)
    else if i ≥ 537 then
12:     return 64.4 + ((537 - i) * 0.16363636363636364)
    else if i ≥ 447 then
        return 66.2 + ((527 - i) * 0.18)
15: else if i ≥ 438 then
        return 82.4 + ((438 - i) * 0.2)
    else if i ≥ 428 then
18:     return 84.2 + ((428 - i) * 0.18)
    else if i ≥ 365 then
        return 86.0 + ((419 - i) * 0.2)
21: else if i ≥ 357 then
        return 98.6 + ((357 - i) * 0.225)
    else if i ≥ 348 then
24:     return 100.4 + ((348 - i) * 0.2)
    else if i ≥ 316 then
        return 102.2 + ((340 - i) * 0.225)
27: end if
    return 109.4 + ((309 - i) * 0.2571428571428572);
```

Group 6

Name: Shunts
Type: Integer
Description: Shunts are small resistors that can be switched in to drain a small amount of energy from one or more of the 96 cells that make up the high voltage battery pack. This is the method used by Nissan to balance the pack by draining energy from the high energy cells. This works because charging stops to prevent overcharging the highest energy cell. So by reducing the energy in the highest energy cell, all the other cells are able to be charged to a higher level. The shunt settings (on or off) for the 96 cell pairs are received as 24 groups of four bits for a total of 12B. The shunt order defines how these four bits (numbered 8,4,2,1) are mapped to the four-cell pairs they are associated with.
Query:
0x79B 02 21 06 00 00 00 00 00
0x79B 30 00 00 00 00 00 00 00
Answer:
0x7BB 10 1A 61 06 **14 55 55 51**
0x7BB 21 **50 55 41 2B 56 54 15**
0x7BB 22 **51** FF FF FF FF FF FF
0x7BB 23 FF FF FF FF FF FF FF
Formula:

Algorithm 5: calculateShunts(i)

```
    SHUNTS_bits = [-1.0] * 24
    SHUNTS_order = "8421"
    for bit in range(12): do
4:      SHUNTS_bits[bit*2] = bytes(G6_payload)[bit] & 0xF
        SHUNTS_bits[(bit*2) + 1] = bytes(G6_payload)[bit] >> 4
    end for
    for i in range(96): do
8:      i3 = SHUNTS_bits[i >> 2]
        j = int(SHUNTS_order[i % 4])
        if (j & i3) != 0 then
            CV_array[i] = -1*CP_array[i]
12:     end if
    end for
```

Group 61

Name: SOH (%)
Type: Integer
Description: State of Health is another indication of the battery's ability to hold

and release energy and is reported as a percentage. When the battery is new SOH=100%
Query: 0x79B 02 21 61 00 00 00 00 00
Answer: 0x7BB 11 4B 61 61 26 9A **25 CA**
Formula: `SOH = (( payload[6] << 8 ) | payload[7] ) / 100`

Group 84

Name: Battery Serial
Type: String
Description: Battery serial number
Query:
0x79B 02 21 84 00 00 00 00 00
0x79B 30 00 00 00 00 00 00 00
Answer:
0x7BB 10 16 61 84 32 33 30 **55**
0x7BB 21 **4B 31 31 39 32 45 30**
0x7BB 22 **30 31 34 38 32 20 A0**
0x7BB 23 00 00 00 00 00 00 00
Formula: `(554b3131393245303031343832 20a0).decode('utf-8')`

7.3.4 Antilock Braking System (ABS)

Name: SWA (°)
Type: Integer
Description: Steering wheel angle. The angle ranges from -720 to 720 degrees with negative being left turn.
Query: 0x740 03 22 12 08 00 00 00 00
Answer: 0x760 05 62 12 08 **AA BB** 00 00
Formula:

```
SWA = ( payload[4] << 8 ) | payload[5];
if (SWA & 32768 == 32768):
    SWA = SWA | -65536;
SWA = SWA / 10
```

Name: Brake
Type: Integer
Description: Brake pedal pressure measured from 0 (not pressed) to 369 (max). Unit is unknown.
Query: 0x740 03 22 12 09 00 00 00 00
Answer: 0x760 05 62 12 09 **AA BB** 00 00
Formula: `Brake = ( payload[4] << 8 ) | payload[5];`

Name: Acc_Pedal
Type: Integer
Description: Acceleration pedal pressure measured from 0 (not pressed) to 200 (max). Unit is unknown.
Query: 0x740 03 22 11 15 00 00 00 00
Answer: 0x760 04 62 11 15 **AA** 00 00 00
Formula: `Acc_Pedal = payload[4]`

7.3.5 Traction Motor Inverter (TMI)

Name: Torque (Nm)
Type: Integer
Description: Motor Torque. Torque is a twisting force that speaks to the engine's rotational force and measures how much of that twisting force is available when an engine exerts itself. Minimum unit step is 0.25 Nm.
Query: 0x784 03 22 12 25 00 00 00 00
Answer: 0x78C 05 62 12 25 **00 00** FF FF
Formula:

```
Torque = ( payload[4] << 8 ) | payload[5];
if (Torque & 32768 == 32768):
    Torque = Torque | -65536;
Torque = Torque / 64.0
```

Name: Motor_Temp (°C)
Type: Integer
Description: This field is the drive motor temperature
Query: 0x784 03 22 11 21 00 00 00 00
Answer: 0x78C 04 62 11 21 **45** FF FF FF
Formula: `Motor_Temp = ( payload[4] - 40 )`

7.3.6 Meter

This single request to METER ECU tells us how the two levers are set. Hence, we can check if the car is turning, the wiper status or the lights status, according to some HEX values.

Query: 0x745 02 21 09 00 00 00 00 00
Answer: 0x765 10 10 61 09 **AA BB CC** XX

```
Wiper = hex(payload[4])
Light1 = hex(payload[5])
Light2 = hex(payload[6])
```

Name: Left_Arrow
Type: Boolean
Description: 0: Off; 1: Blinking - the car is turning left
Formula: `(Wiper == '0x22' or Wiper == '0x9') and (Light1 == '0x6')`

Name: Right_Arrow
Type: Boolean
Description: 0: Off; 1: Blinking - the car is turning right
Formula: `(Wiper == '0x22' or Wiper == '0x9') and (Light1 == '0xa')`

Name: Rain_Level
Type: Integer
Description: Rain sensitivity level for front wiper (automatic mode) (1: low, 4: high)
Formula:

```
if(Wiper == '0xf' or Wiper == '0x1f' or Wiper == '0x47' or Wiper == '0x87'); Rain_level = 1
if(Wiper == '0xd' or Wiper == '0x1d' or Wiper == '0x45' or Wiper == '0x85'); Rain_level = 2
if(Wiper == '0xb' or Wiper == '0x13' or Wiper == '0x43' or Wiper == '0x83'); Rain_level = 3
if(Wiper == '0xa' or Wiper == '0x19' or Wiper == '0x41' or Wiper == '0x81'); Rain_level = 4
```

Name: Front_wiper_level
Type: Integer
Description: -1: Off; 0: Automatic; 1: Continuous slow speed; 2: Continuous high speed
Formula:

```
if(Wiper == '0xf' or Wiper == '0xd'); Front_wiper_level = -1
if(Wiper == '0xb' or Wiper == '0xa'); Front_wiper_level = -1
if(Wiper == '0x1f' or Wiper == '0x1d'); Front_wiper_level = 0
if(Wiper == '0x13' or Wiper == '0x19'); Front_wiper_level = 0
if(Wiper == '0x47' or Wiper == '0x45'); Front_wiper_level = 1
if(Wiper == '0x43' or Wiper == '0x41'); Front_wiper_level = 1
if(Wiper == '0x87' or Wiper == '0x85'); Front_wiper_level = 2
if(Wiper == '0x83' or Wiper == '0x81'); Front_wiper_level = 2
```

Name: Front_washer
Type: Boolean
Description: 0: Not active; 1: Active
Formula: `(Wiper == '0x22' or Wiper == '0x9') and (Light1 == '0x1')`

Name: Rear_washer
Type: Boolean
Description: 0: Not active; 1: Active

Formula: `(Wiper == '0x22' or Wiper == '0x9') and (Light1 == '0x31')`

Name: Rear_wiper_level
Type: Integer
Description: -1: Off; 1: Intermittent; 2: Continuous low speed
Formula:

```
if(Light1 == '0x41'); Rear_wiper_level = 1;
if(Light1 == '0x91'); Rear_wiper_level = 2;
```

Name: Position_lights
Type: Integer
Description: -1: Off; 0: Automatic mode; 1: On
Formula:

```
if(Light1 == '0x02' and (Light2 == '0x20' or Light2 == '0x28')); Position_lights = 0
if(Light1 == '0x02' and (Light2 == '0x00' or Light2 == '0x08')); Position_lights = 1
if(Light1 == '0x03' and Light2 == '0x08'); Position_lights = 0
if(Light1 == '0x03' and Light2 == '0x00'); Position_lights = 1
```

Name: Headlights
Type: Integer
Description: -1: Off; 0: Automatic mode; 1: On
Formula:

```
if(Light1 == '0x00' and (Light2 == '0x10' or Light2 == '0x18')); Headlights = 0
if(Light1 == '0x00' and (Light2 == '0x30' or Light2 == '0x38')); Headlights = 0
if(Light1 == '0x01' and (Light2 == '0x10' or Light2 == '0x18')); Headlights = 0
if(Light1 == '0x02' and (Light2 == '0xc0' or Light2 == '0xc8')); Headlights = 1
if(Light1 == '0x02' and (Light2 == '0xe0' or Light2 == '0xe8')); Headlights = 1
if(Light1 == '0x03' and (Light2 == '0xc0' or Light2 == '0xc8')); Headlights = 0
```

Name: Anti_fog_lights
Type: Boolean
Description: -1: Off; 1: On
Formula:

```
if(Light1 == '0x00' and (Light2 == '0x18' or Light2 == '0x38')); Anti_fog_lights = 1
if(Light1 == '0x01' and Light2 == '0x18')); Anti_fog_lights = 1
if(Light1 == '0x02' and (Light2 == '0x08' or Light2 == '0xc8')); Anti_fog_lights = 1
if(Light1 == '0x02' and (Light2 == '0x28' or Light2 == '0xe8')); Anti_fog_lights = 1
if(Light1 == '0x03' and (Light2 == '0x08' or Light2 == '0xc8')); Anti_fog_lights = 1
```

Name: HighBeam
Type: Integer
Description: -1: Off; 0: Automatic mode; 1: On; 2: Manual
Formula:

```
if(Light1 == '0x03' and (Light2 == '0x00' or Light2 == '0x08')); HighBeam = 0
if(Light1 == '0x01' and (Light2 == '0x10' or Light2 == '0x18')); HighBeam = 1
if(Light1 == '0x03' and (Light2 == '0xc0' or Light2 == '0xc8')); HighBeam = 1
if(Light1 == '0x00' and (Light2 == '0x30' or Light2 == '0x38')); HighBeam = 2
if(Light1 == '0x02' and (Light2 == '0x20' or Light2 == '0xe0')); HighBeam = 2
if(Light1 == '0x02' and (Light2 == '0x28' or Light2 == '0xe8')); HighBeam = 2
```

7.3.7 Heating, Ventilation and Air Conditioning (HVAC)

Name: HeaterTemp
Type: Integer
Description: Heater temperature from 16.5°C to 30°C set by the driver. Apparently this value increases/decreases accordingly to the heater's UP/DOWN button, but the values interval sometimes shifts. Further analysis are required to check how to convert the HEX values to the range 16.5-30.
Query: 0x744 02 21 10 00 00 00 00 00
Answer:
0x764 10 36 61 10 30 3b 33 0d
0x764 21 2f 3b 00 32 0e 00 80
0x764 22 84 **8D** 00 00 00 00 00
Formula: `HeaterTemp = payload[2];`

Name: FanSpeed
Type: Integer
Description: Fan speed from 0 (off) to 8 (max)
Query: 0x744 02 21 10 00 00 00 00 00
Answer:
0x764 10 36 61 10 30 3b 33 0d
0x764 21 2f 3b 00 32 0e 00 80
0x764 22 **84** 8D 00 00 00 00 00
Formula: `FanSpeed = payload[1] - 131 ;`

7.4 Evaluation

Thanks to the collaboration with two *HVAC systems* students[3] from the University of Bologna, we performed a three-day driving test on Bologna streets, driving a Nissan Leaf 2018. We analyzed its energy consumption (EC) and other parameters such as the energy recovered by the regenerative braking (RB), comparing different

[3]Jacopo Ferretti and Francesco Romani

driving styles. Furthermore, we checked the HV battery current/voltage, and how the e-Pedal/ECO tools impact range and speed.

7.4.1 Experimental setup

Driving cycles

Driving tests were performed along the ring roads of Bologna, which is an 8km round circuit around the city center. Along the ring road, there are some climbs and descents, many traffic lights, and it is one of the most congested roads throughout the city.

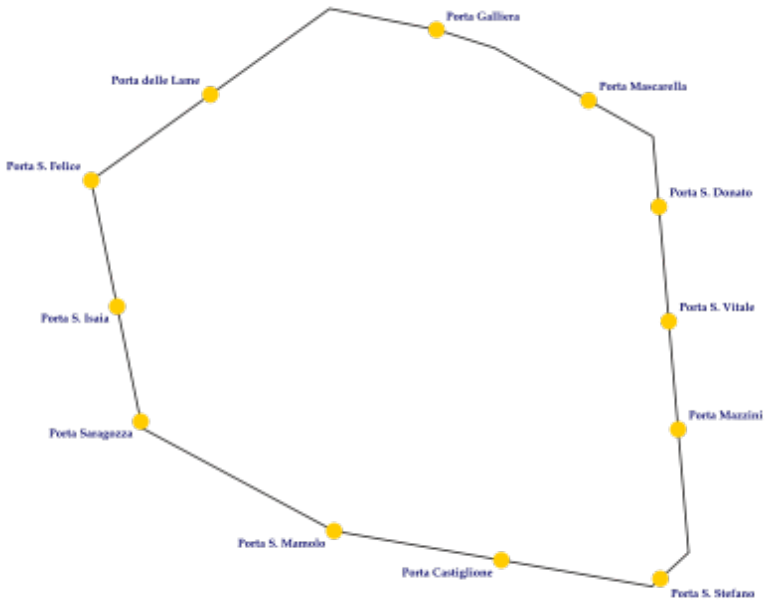

Fig. 7.4: Ring road of Bologna

In two weeks we had three afternoons test sessions; overall we performed 18 laps. Unfortunately, some records of two laps were lost due to cellular connection issues and we could not send any data to the server.

Date	Laps	ECO	e-Pedal	Driving style	HVAC
29th October 2020	2	✓	✓	Slow	×
	3	✓	✓	Slow	✓
5th November 2020	3	✓	✓	Slow	✓
12th November 2020	1	✓	×	Slow	✓
	3	×	✓	Fast	×
	2	×	×	Fast	×
	1	×	×	Slow	×
	1	✓	×	Slow	✓

Table 7.3: Driving test cycles

7.4.2 Data analysis

Voltage and temperature

Two important parameters of a battery electric vehicle (BEV) accumulators are the temperature and the battery pack's voltage: they both need to be monitored constantly by the Battery Management System (BMS) to prevent very dangerous situations like a thermal runaway. The thermal runaway is considered one of the major safety issues related to a BEV; due to the great number of cells installed in the car, if some of them are faulty, they could set a violent fire that may lead to explosions [113]. Battery voltage is also a crucial parameter: one overvoltaged cell may be a symptom of unexpected stress that could lead to a fire. On the other hand, if a cell undervoltages, it means that the chemistry inside is deteriorating, reducing life expectancy. In fig. 7.5, fig. 7.6, and fig. 7.7 different plots of the pack voltage and temperature can be seen.

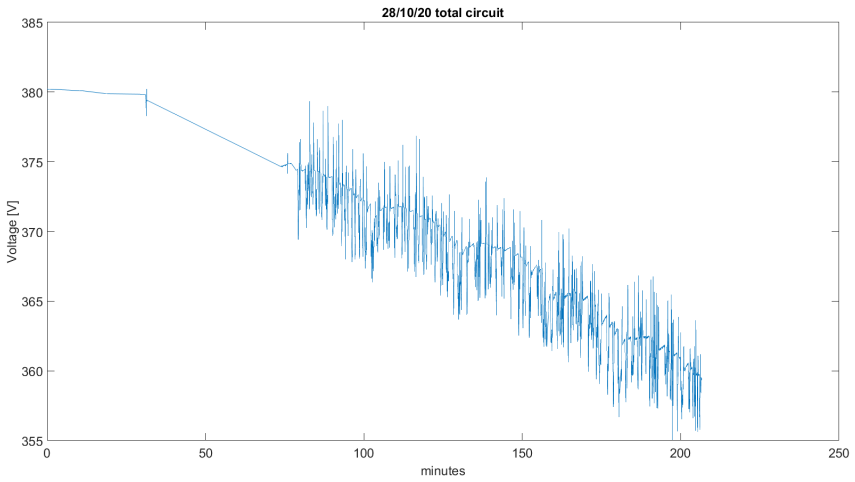

Fig. 7.5: HV battery voltage - 28^{th} October 2020 Lap

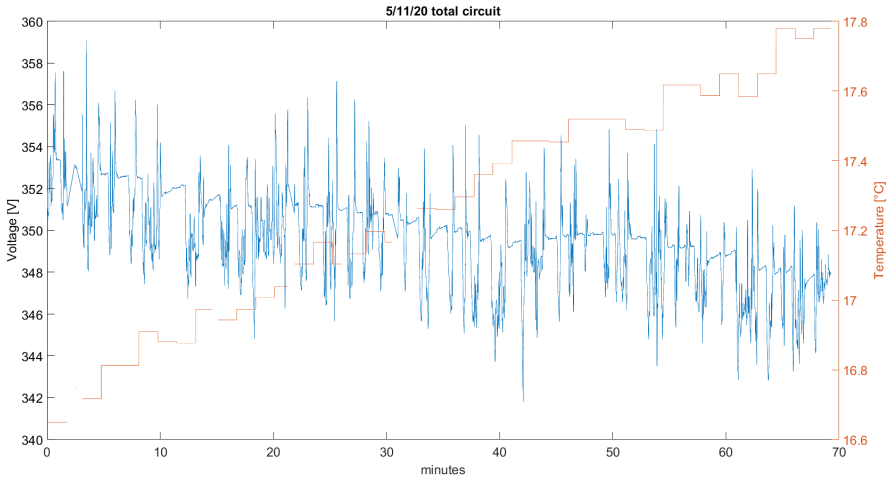

Fig. 7.6: HV battery voltage and temperature - 5^{th} October 2020 Lap

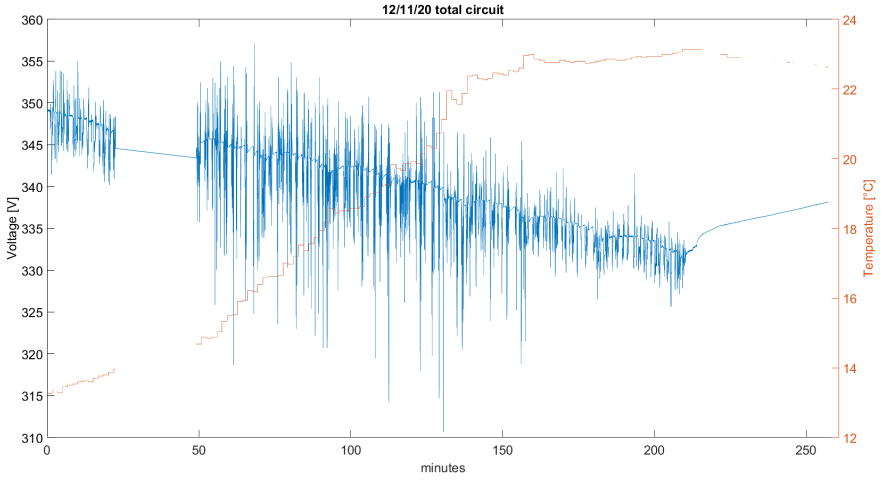

Fig. 7.7: HV Battery Voltage and temperature - 12^{th} November 2020 Lap

From above, we can claim that the voltage's plateaus are at different levels: since the 29/10/20 test, the battery energy has decreased, and the car was never recharged. Thus, the voltage difference between the first lap of the first day and the last lap of the last day is around 40V (from 375V to 335V), more than the 10% of the maximum voltage.

Motor energy consumption

In this analysis we included every lap different characteristic; we checked which driving cycle had the lower motor energy consumption (EC), the best energy recovered from the regenerative braking (RB), and we compared them with the specific

energy consumption (kWh/km). Eventually, we could claim which was the most energy-saving driving style. The results are shown below:

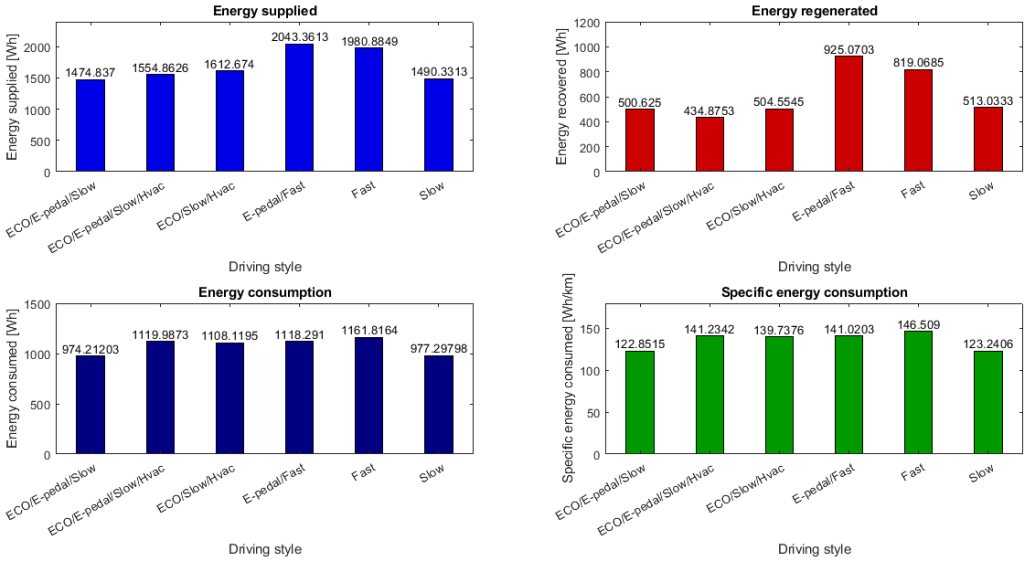

Fig. 7.8: Energy plots with different driving styles

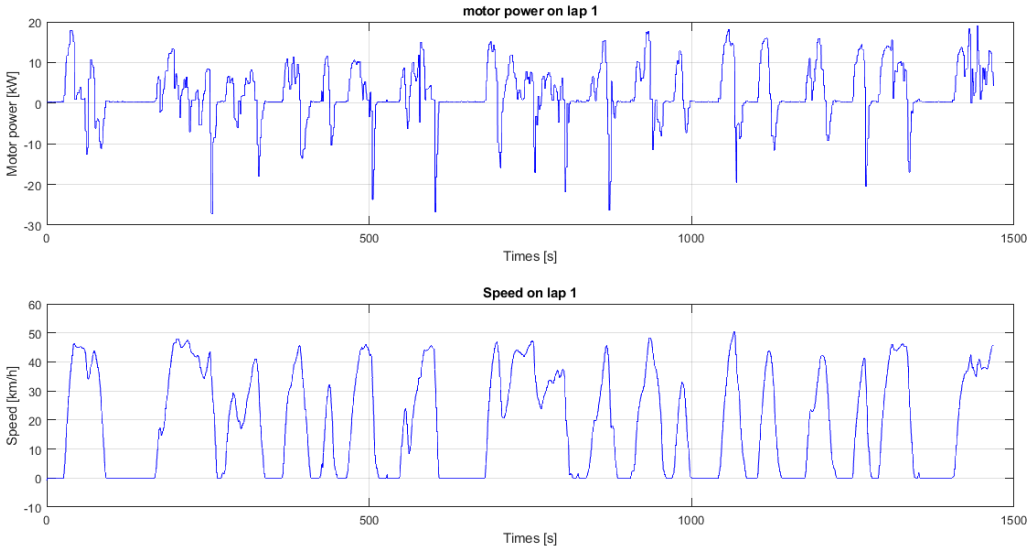

Fig. 7.9: Motor Power and Speed profiles - Slow lap

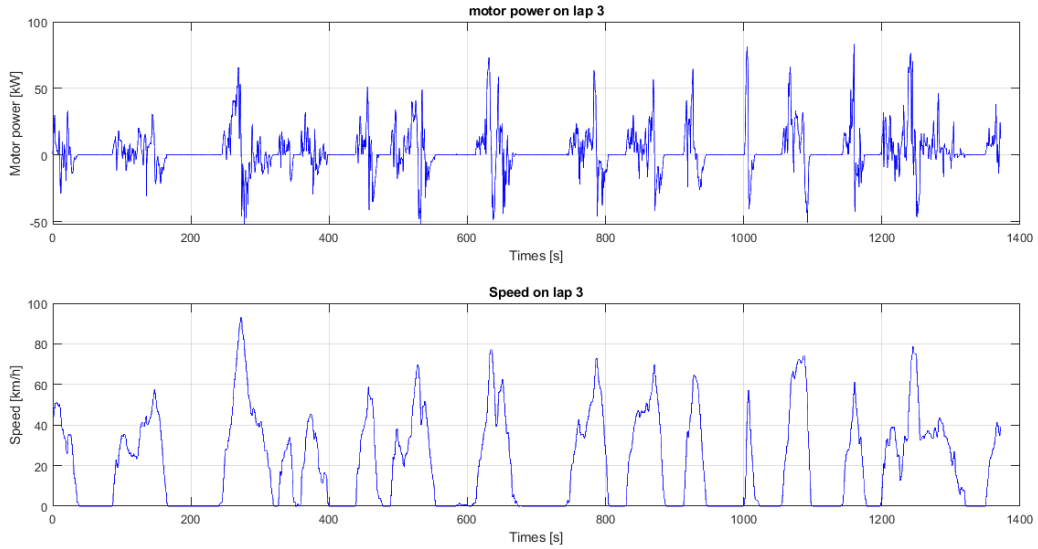

Fig. 7.10: Motor Power and Speed profiles - Fast lap

The fig. 7.8 is made up of four different histograms: (1) the total energy supplied by the battery pack, (2) the total energy recovered due to the regenerative braking system (which works whether the e-Pedal is activated or not), (3) the energy balance between energy supplied/energy regenerated, (4) the specific energy consumption.

- Comparison between columns 2 (ECO/e-Pedal/Slow/HVAC) and 3 (ECO/Slow/HVAC)

	E_c [Wh]	%	E_r [Wh]	%	E_s [Wh/km]	%
ECO, e-Pedal, HVAC, slow	1555	/	435	/	141	/
ECO, HVAC, slow	1612	3.67	505	16.09	140	-0.71

Table 7.4: Comparison between columns 2 and 3 - Energy comparison #1

- The use of the e-Pedal impacts the EC. The e-Pedal enhances the RB of the car. When the e-Pedal is working, the RB system starts as soon as we release the acceleration pedal, otherwise, it works when the brake pedal is pressed.
- There is no big difference between columns 2 and 3 in all the plots; the car speed is quite slow;
- The energy supplied increases by +3.67% in the column 3;

- The regenerative energy increases significantly, even if the e-Pedal is off (+16.09%);
- From the last plot, the specific energy consumption of the two cycles is slightly the same (around 140Wh/km).
- At lower speed, the presence of the e-Pedal is irrelevant.

- Comparison between columns 1 (ECO/e-Pedal/Slow) and 4 (e-Pedal/Fast)

	$\mathbf{E}_c$ **[Wh]**	%	$\mathbf{E}_r$ **[Wh]**	%	$\mathbf{E}_s$ **[Wh/km]**	%
ECO, e-Pedal, slow	1475	/	501	/	123	/
e-Pedal, fast	2043	38.51	925	84.63	141	14.63

Table 7.5: Comparison between columns 1 and 4 - Energy comparison #2

- The use of the ECO mode reduces the power requested by the motor and influences the speed too;
- The energy supplied for a complete lap is much higher in the faster lap than in the slower one (+38.51%).
- The recovered energy from the brakes in the faster lap is almost twice the recovered energy of the slower one (+84.63%). This peculiarity ensures that the total EC of the faster lap is not too high, respect with the slower one;
- The eco-cycle has the specific energy consumption of 123Wh/km while the fast cycle with the e-Pedal is 141Wh/km (+14.63%);
- Since the battery has 40kWh total capacity, the variation above is translated into an autonomy difference of almost 42km (325km for the first cycle and 283km for the second one).

- Comparison between columns 4 (ECO/e-Pedal/Slow) and 5 (e-Pedal/Fast)

	$\mathbf{E}_c$ **[Wh]**	%	$\mathbf{E}_r$ **[Wh]**	%	$\mathbf{E}_s$ **[Wh/km]**	%
e-Pedal, fast	2043	/	925	/	141	/
fast	1980	-3.08	819	-11.46	147	4.26

Table 7.6: Comparison between columns 4 and 5 - Energy comparison #3

- The only difference is e-Pedal presence;

- If the e-Pedal is active, the total energy by the battery is quite higher than the lap without it (+3.08%); this may be either due to different traffic conditions or to a minimal change of the driving style;
- With e-Pedal ON, there is a significant increase in the energy recovered, due to the RB, in comparison with the other driving cycle (+11.46%);
- From the last plot, the specific energy consumption of the two cycles is not so different (141Wh/km the former and 147Wh/km the latter);
- The autonomy gain is just about 12km more for the cycle with e-Pedal on.

- Comparison between columns 1 (ECO/e-Pedal/Slow) and 6 (Slow)

	$\mathbf{E}_c$ **[Wh]**	**%**	$\mathbf{E}_r$ **[Wh]**	**%**	$\mathbf{E}_s$ **[Wh/km]**	**%**
ECO, e-Pedal, slow	1475	/	501	/	123	/
slow	1490	1.02	513	2.40	123	0.00

Table 7.7: Comparison between columns 1 and 6 - Energy comparison #4

- Same cycles. The former one has Eco-mode and the e-Pedal on, while the latter has both the pedals OFF.
- The first cycle has a lower energy consumption (-1.02%) and lower energy recovered (-2.40%).
- These two cycles have the same specific energy consumption (123Wh/km).
- Power saving tools are not so useful if used in low-speed cycles.

- Comparison between columns 1 (ECO/e-Pedal/Slow) and 5 (Fast)

	$\mathbf{E}_c$ **[Wh]**	**%**	$\mathbf{E}_r$ **[Wh]**	**%**	$\mathbf{E}_s$ **[Wh/km]**	**%**
ECO, e-Pedal, slow	1475	/	501	/	123	/
fast	1981	34.31	819	63.47	147	19.51

Table 7.8: Comparison between columns 1 and 5 - Energy comparison #5

- They are two opposite driving cycles. The slower uses all the power-saving tools (ECO and e-Pedal) while the faster doesn't.
- In the faster lap, the energy supplied by the battery pack is greater than 33% (+34.31%).

- From the second plot, the energy recovered thanks to the RB is higher in the faster lap (+63.47%).

- On energy balancing, there is the highest gap between all comparisons (123Wh/km vs. 147Wh/km), which can be translated into 53km of additional autonomy for the slower cycle.

Overall, the power-saving tools, such as the ECO-mode and the e-Pedal, are really useful only if the driving style is fast.

The higher the speed of the cycle, the higher will be the energy supplied by the battery; a higher speed also leads to higher energy recovered by the regenerative braking, even if the e-Pedal is switched off.

To save energy and to achieve longer distances, the slower speed is the only way; faster driving has better results if the e-Pedal is on, even if it does not change any performance of the car; the e-Pedal provides higher autonomy even if it will be quite small.
A faster drive implies higher charge and discharge of the HV battery, and this can eventually reduce the battery SOH, and hence, its life.

Path analysis

In this test, we compare the average speed and the average driving time for every driving cycle. *Is fast driving helpful to time-saving? Does it increase the energy consumption?* In the histograms below we can see respectively the average speed (total and while driving), the time while moving, when we are stopped, and lastly, the total time spent for a lap.

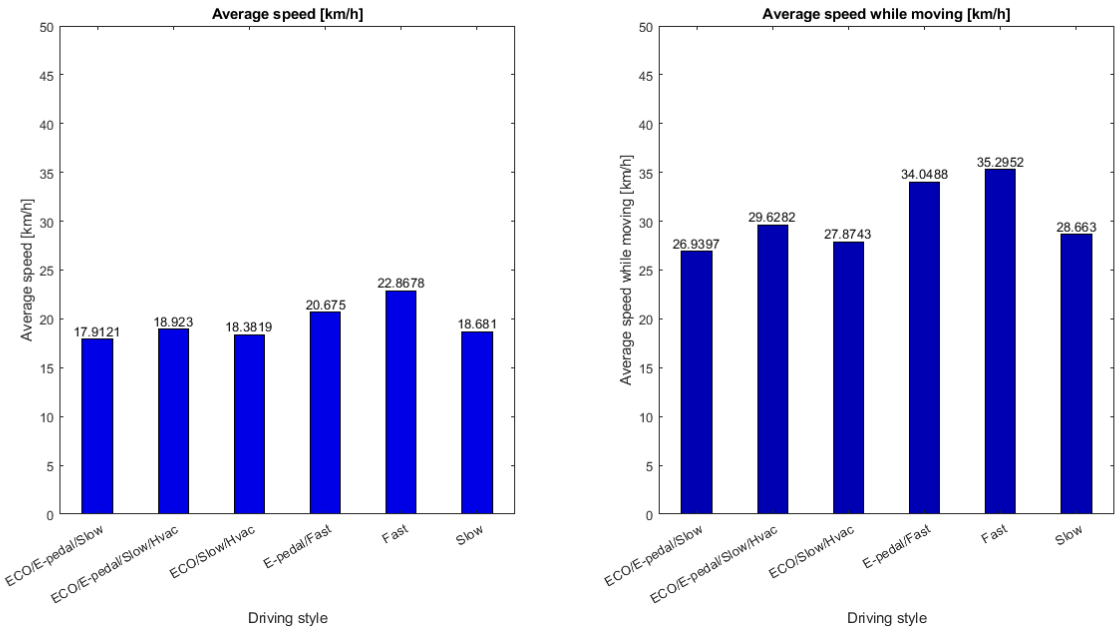

Fig. 7.11: Average speed

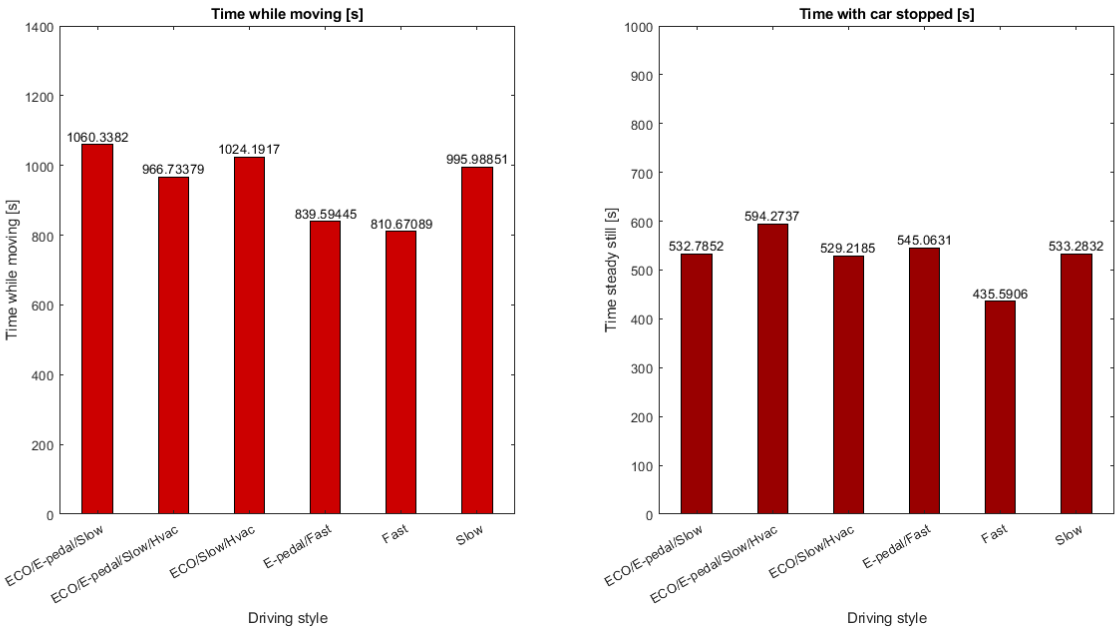

Fig. 7.12: Average time moving and steady still

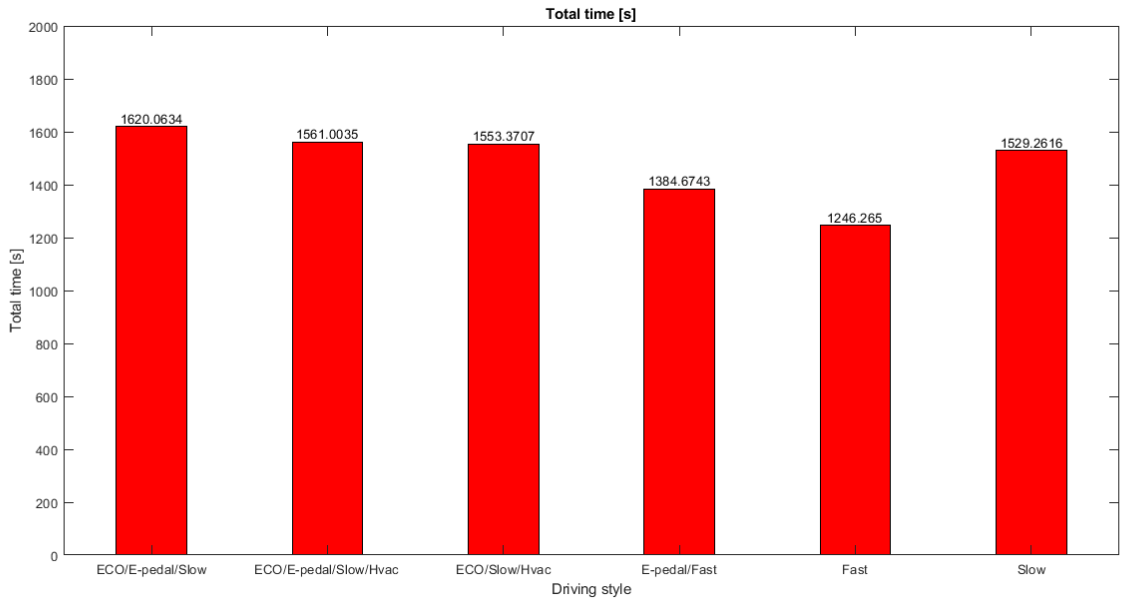

Fig. 7.13: Total time for a lap

	Slow	**Fast**	**%**
$\mathbf{v}_{avg}$ **[km/h]**	18,5	22	18.92
$\mathbf{v\_moving}_{avg}$ **[km/h]**	28	37,5	23.21
$\mathbf{t}_{moving}$ **[s]**	1020	830	-18.63
$\mathbf{t}_{stop}$ **[s]**	550	480	-12.73
$\mathbf{t}_{total}$ **[s]**	1570	1310	-16.56

Table 7.9: Comparison between slow and fast cycles - Time percentages

From the first three columns, we see that the total time is almost the same. Indeed, there could be some differences due to different traffic conditions (laps have been recorded on different days and at different times).

If we drive slowly, it takes around 26 minutes to complete a lap, with an average speed of 18.5km/h, including the minutes when the car is stopped; if we do not include them, the average speed goes up to 28 km/h. From our data, we noticed that we spent almost 35% of our time steady still, which corresponds to 9 minutes each lap.

The average time from columns 4 (e-Pedal/Fast) and 5 (Fast) is around 22 minutes, which is 18% less of the total time by slower runs. Surprisingly, the steady time is still high: 8 minutes; the driving time decreases of almost 20% and the average moving speed goes from 28km/h to 34.5km/h.

Including the stopping time, which is almost the same as in the slower laps, the average speed goes down to 22km/h; which is a small but relevant difference than

the slow rides (+19%).

Now it is possible to compare the time saved while driving faster, and see if the extra energy consumed was worth it. Below the comparison between the two opposite driving cycles.

	Slow	**Fast**	**%**
$\mathbf{v}_{avg}$ **[km/h]**	17.91	22.87	27.69
$\mathbf{t}_{moving}$ **[s]**	1060	811	-23.49
$\mathbf{t}_{stop}$ **[s]**	533	436	-18.20
$\mathbf{t}_{total}$ **[s]**	1620	1246	-23.09
$\mathbf{E}_s$ **[Wh/km]**	123	147	19.51

Table 7.10: Comparison between columns 1 and 5 - Time and energy

From the energy analysis, there is a difference in consumption by almost 20% in the fastest lap respect than the slowest one; on the other hand, the time saved is around 23%.

Nevertheless, if we take every absolute value into account, we need to evaluate if worth saving 6 minutes every day - in a total trip of 20-25 minutes - losing at least 50km of range too. The answer should be no.

Besides, in this driving tests, we drove on urban streets, where the energy consumption of an EV is lower; in an extra-urban scenario, the energy consumed would be even higher.

In the end, a slower driving style is the recommended style for the following reasons:

- It increases the range of, at least, 50km;
- Less battery stress, thanks to lower currents;
- Less mechanical stress and less deterioration of mechanical brakes;
- Less risk of fine and road accident.

Chapter 8

X-Fi: Revisiting WiFi Offloading in the Wild for V2I Applications

8.1 Introduction

Vehicle to Internet (V2I) communication has been a topic of significant research over the years. New systems that improve V2I communication [114, 115] and measurement studies of how commercial and research systems fare under the prevalent network conditions have also described the evolution of such systems [116, 117]. Many efforts observed that if every vehicle were to connect and communicate with Internet-hosted infrastructure, the aggregate data volumes are likely to overwhelm common cellular capacities. Therefore, a common thread of work has also been to explore *opportunistic WiFi offloads*, often using open WiFi Access Points (APs). Cabernet [116], circa 2008, is a good exemplar of such a system where various location-related data from Boston taxis were uploaded using such opportunistic encounters with such open WiFi APs. The authors conducted an extensive measurement study that demonstrated the feasibility of such a system and for their region of interest, achieved a mean throughput of 86Kbps.

The last decade has seen dramatic changes in the way WiFi networks are available in large metropolitan areas as well as in the connected cars applications. In many cases, wireless providers have begun to offer a WiFi service to its paid customers through APs mounted across street poles, and backed over fiber, or point-to-point wireless links. This allows users to sometimes have widespread data services across the entire city. Unlike opportunistic and open WiFi networks, these use *authentication* for access, and communication occurs over *encrypted* data channels. Further, these networks are deployed in a structured manner by the operator to maximize coverage across the city. Such structured and authenticated WiFi networks with encrypted data paths have significant implications in delays for access, and ability for "handoffs" impacting the use of WiFi offloads in vehicular settings.

In parallel to these developments in the wireless world, significant changes have emerged in vehicular systems and applications. Vehicles are more and more be-

coming large data *prosumers*. There is the vision of fully autonomous vehicles on the horizon (Level 5 automation [118]), with major trials being led by big players (Waymo, Uber, etc.) in select markets. Intel speculates each autonomous vehicle will produce about 4TB of data per day, mostly from the LiDAR and the cameras [119]. We conducted our own tests on the field with 30+ 2018 Nissan Leaf vehicles that form a part of a university's car-sharing fleet. On an average day (˜10h), each car collects and streams 20MB from the internal engine control units. To mimic an autonomous vehicle, we instrumented one car with sensors including 4 sonar(˜540Kbps), a front MRR radar(˜800Kbps), 4 (8MP, 60fps, 16bit depth) cameras[1] (˜3.84Gps), and a 2D LiDAR[2] (˜270Mbps), figure that exponentially grows if using a 3D LiDAR). Although the actual deployment of autonomous vehicles at scale is still a few years out, various forms of assisted driving systems are gaining in popularity. Systems such as Tesla Autopilot (updated ˜5 times a year, requires the download of 500MB) [120, 121, 122, 123] and other Internet-connected Advanced Driver Assistance Systems (ADAS) perform continuous data collection from vehicles on the road (˜5.6Gbps), aggregate them centrally, and learnings from them are shared back with the vehicles again. In addition, these connected vehicles perform numerous data exchanges with cloud-hosted services, e.g., for over-the-air firmware updates, such as BMW [124], to support Connected Battery Management Systems (BMS) [125], and for various AI-based driving functions.

Today's approaches for connected vehicles: At present, connected vehicles exclusively rely on cellular networks. However, it is apparent that WiFi offloading provides benefits, more than ever before, to all the stakeholders in the connected-vehicle arena: **(1)** cellular networks are expensive for large data users; **(2)** mobile network operators suffer from such users and, even in their most expensive plans, reduce data rates past a given data-consumption threshold[3]; **(3)** adding millions of high demanding vehicles to the network will force operators to make massive infrastructure investments [126].

[1]Tesla mount 8 cameras but no LiDAR

[2]Low-end Hokuyo UST-30LX

[3]Reducing the data rate for a specific user is a common practice among Mobile Network Operators (MNOs). For instance, in the US, high-end AT&T and T-Mobile plans have a data cap per billing cycle of 50GB and 100GB, respectively. Once a user reaches the data cap, their service speeds are reduced.

Operator	Monthly Cost	Unrestricted Usage	Data Cap	Throughput Claimed (Mbps)	Throughput Sensed (Mbps)
AT&T (USA)	85 USD	✗	>22GB	at least 10/1	27.5/6
T-Mobile (USA)	75 USD	✗	>33GB	usually 5–31/3-15	25.8/8.6
Verizon (USA)	70 USD	✗	>75GB	940/880	25.3/7.9
CTM (Macao)	60 USD	✗	>10GB	112/37.5	35.5/30.94
Vodafone (IT)	40 €	✓	-	1200/150	24.3/7
TIM (IT)	50 €	✗	>100GB	2000/300	24.1/7.6
Wind (IT)	12 €	✗	>100GB	1000/75	21.6/7.5
Free (FR)	20 €	✗	>100GB	Unclaimed	14.2/5.2

Table 8.1: Mobile network operators prices and limits [127]. Throughput is throttled to 128Kbps by many providers if data cap is reached; Italian providers stop any service if consume >100GB/month.

***X-Fi* and WiFi offloads:** We believe that offloading part of V2I traffic to WiFi would be a critical part of the future, in which vehicles roam onto provider-managed WiFi infrastructures that are increasingly getting deployed across major metros, Provider-managed WiFi services are marketed as a perk attached to the home-Internet contract and, therefore, included in its cost. Usually, users can access their operator's WiFi nationwide, thus benefiting from this flexibility everywhere. Table 8.1 shows a summary of MNO costs in Europe, USA, and Macao. Reading the fine print, we observed many MNOs curb users consuming excessive traffic by shaping their bandwidth or cutting their access. The on-going transition toward 5G [128, 129] and the deployment of future WiFi standards (i.e. WiFi 6) [130] are not going to change the current cost-benefits ratio. 5G cells will trade high performance at the price of coverage. Besides, 5G architecture is designed to natively integrate multi-radio access, and WiFi is one of them [131, 132]. Therefore, it is reasonable assuming 5G cellular access will have a per-user cost similar to LTE, with WiFi remaining a free perk of home network providers.

This chapter introduces: **(a)** *X-Fi*, a system that enables a moving vehicle to connect to provisioned, encrypted, WiFi networks, proven to work across multiple countries and operators; **(b)** *X-Perf*, a tool to measure TCP performances in intermittent networks; **(c)** an extensive study of V2I WiFi-offloading feasibility in-the-wild, across 4 cities and 3 continents, covering more than 4250Km, over a period of 14 months.

Differently from the previous seminal efforts carried out several years ago (Table 8.2), our WiFi offloading study today has in a very different networking context. The communication landscape is changed: open WiFi networks **no longer exist**, today's WiFi coverage is **operator provisioned**, **encrypted**, and, therefore, **requires authentication**.

In a nutshell, the most important lessons we learned from our study are: **(1)** ultimately, it is possible to use operator managed encrypted WiFi networks to support applications running in moving vehicle; **(2)** the 1GB/hour throughput achieved by

X-Fi is an order of magnitude higher than what observed in previous studies, thus providing a remarkable data-bill relief for non-real-time bulk automotive applications. This is further borne out by the fact that WiFi is an integral part of 5G, which, in turn, will benefit from offloading onto existing, cheaper infrastructure; **(3)** as a guideline for those network operators willing to provide support vehicular application, as observed in LA, outdoor deployed access points have shown substantially better performance; **(4)** the bottleneck to a successful connectivity establishment is the DHCP procedure. To address that, operators grant extended lease to the same IP address, thus saving time negotiating a new IP address at every new connection; **(5)** the ratio of successful IP acquisitions over link-layer associations varies across different cities, however, the percentage of successful TCP connections given the acquisition of an IP address is similar. Hence, once a vehicle obtains an IP, it is likely to set up a TCP connection regardless the infrastructure.

Summary: this chapter has two main components: a systems design (*X-Fi*), resulting in a toolkit for intermittent connectivity, and a measurement component (based on *X-Perf*), aiming at characterizing commercial WiFi offloading opportunities for **non-real-time bulk transfer applications** in urban scenarios across different continents and network infrastructures.

8.2 Related Work

Using WiFi to provide moving vehicles with Internet access has been studied by several in the past. A more in-depth technical comparison between our work and previous art is reported in Table 8.2.

Accessing the network: Ott et al. [133], confirmed a vehicle can connect with individual WiFi APs placed along the roadside. Gass et al. [134] studied the impact of the back-haul on WiFi based V2I connections. Hadaller et al. [135] carried out similar experiments focusing on the back-haul impact on end-to-end performance. Besides, they studied the performance of stock WiFi clients in WiFi based V2I identifying the limits of WiFi Stack implementations providing guidelines. *While [133, 134, 135] focus on connecting to individual AP in a controlled testbed, our system focuses on deployment-ready WiFi-based in-the-wild off-loading using commercially available WiFi networks.* Mahajan et al. [136] conducted a measurement campaign on a controlled V2I WiFi testbed, finding that the connectivity offered by individual AP is often spoiled by a "gray" period during which its quality is very poor. However, they only analyzed link-layer metrics, without providing end-to-end performance results. Soroush et al. [137] reported V2I connectivity results based on their experience operating a metro-scale V2I WiFi testbed. They did not provide any end-to-end performance results.

V2I Performance Evaluation: Bychkovsky et al.[138] ran the first measurement study on vehicular Internet access using in-situ open, unencrypted, WiFi

networks in Boston. They confirmed vehicles can upload hundreds of KB per WiFi association with unauthenticated and unencrypted APs. However, that is not applicable today as: **(1)** open WiFi networks have almost disappeared in urban areas [139]; **(2)** their measurement results are based on HostAP-MadWiFi, companion that used low-level features which no longer exists. Deshpande et al. [140] measured the TCP performance of vehicular WiFi connectivity in-the-wild. Similarly to [138], they focus on open WiFi networks using MadWiFi, now discontinued. Limiting the server side TCP connection timeout to 1s, they reduced the effects of WiFi intermittent coverage. *Our work is based on standard WPA-Supplicant. X-Fi supports V2I using authenticated and encrypted commercial WiFi networks. Similarly to [140] we performed extended measurements in several metro-areas, however the 3G/WiFi comparison is out of this paper scope.* Eriksson et al. [116] propose Cabernet: a WiFi client to establish fast connectivity with transient APs from moving vehicles; and an enhanced transport protocol to cope with intermittent connectivity. Their experiments were in-the-wild with open WiFi networks, and show the capability of open WiFi networks to provide Internet access to moving vehicles. However, Cabernet has several limitations: **(1)** requires the client WiFi card to be in **monitor mode**[4]. This approach would incur substantial performance issues in today's urban scenarios as the AP density on the same channel is such to require a lot of CPU cycles to process all incoming packets; **(2)** rely on low-level features as *radiotap* that limits software portability; **(3)** employs a custom transport protocol, thus requiring changes to the applications; **(4)** does not work at all if **authentication** or **encryption** are enabled, as in current commercial WiFi networks. Soroush et al. [141] proposed to maintain concurrent associations with different APs to enhance performance and reliability of V2I WiFi connectivity. They concluded that throughput improvements can only be achieved if mobile WiFi clients associate with multiple APs on the same radio channel based on the assumption that the bottlenecks are the paths between APs and servers. However, from our results, we can see that the bottlenecks more likely lie in the wireless part (achieved goodput is less than the typical broadband bandwidth). Further, their methods may not work under today's circumstances, and their evaluation based on unplanned open WiFi networks.

Recent studies by Lee et al. [139] show there are virtually no open access points: the majority of networks are protected using either a web-based captive portal, or the WPA/WPA2 standards with a pre-shared key (PSK), or an EAP-based method [142].

In contrast, X-Fi is integrated in the WPA-Supplicant standard code flow and

[4]We confirmed this with the authors. Cabernet source code can be found at: `http://people.csail.mit.edu/stoledo/release/quickwifi/`

	Date	*WiFi deployment*	*AP 802.11 Standard*	*Exp. Conditions*	*Evaluation scale*	*Code or data released?*	*Goodput UP/DOWN (Kbps)*	*Long-term goodput * UP/DOWN (MB/h)*	*Median E2E conn. setup delay*	*Median assoc. T*	*Mean assoc. gap*	*Median E2E conn. T*	*Median E2E conn. gap*	*Mean E2E conn. gap*
Balasubramanian et al. [150]	2010	UO APs + Planned TB APs	b	TB + ITW	3C (US)	None	∼200/∼280	-	-	-	-	-	-	-
Soroush et al. [137]	2004∼08	UO APs + Planned TB APs	abg	TB + ITW	1C (US)	Data only	-	62/90	-	∼9s	-	-	-	-
Eriksson et al. [116]	2008	UO APs	bg	ITW	1C (US)	Code only	-/∼320	-/38	289ms	-	-	∼4s	32s	126s
Bychkovsky et al. [138]	2006	UO APs	b	ITW	1C (US)	None	240/-	2.67/-	12.9s	∼24s	75s	∼13s	10s	260s
Hadaller et al. [135]	2007	One open AP	abg	TB	One AP + One car TB	None	-	-/-	-	-	-	-	-	-
Soroush et al. [141]	2011	UO APs	abg	TB	2C (US)	Code only	2400/-	427/-	∼1.4s	-	-	∼12s	∼60s	-
Deshpande et al. [140]	2010	Commercial APs	bg	ITW	1C (US)	None	-/300	-/22	-	-	-	∼2.5s	∼3.5	-
X-Fi: Macao	2019∼20	Commercial APs	n/ac	ITW	4C (US/ASIA/EU)	Both	9787/5338	1322/1028	1.4s	8s	111s	9s	31s	124s
X-Fi: LA							2764/4338	888/2103	2.1s	17s	33s	22s	12s	43s
X-Fi: Bologna							1797/1159	1470/1250	0.9s	9s	12s	10s	5s	18s

Table 8.2: Comparison with previous work (UO=Unplanned Open; TB=Testbed; ITW=in the wild; C=Cities)

relies on signal-based and database-based algorithms to manage the connections (§8.3.1). X-Fi has been evaluated on commercial WiFi deployments in 4 cities across 3 continents downloading, on average, over 1GB in one hour drive. X-Fi and X-Perf measure realistic V2I application-performances on intermittent networks using off-the-shelf TCP.

Custom Transport Layer Protocols: Several studies [143, 116, 144] proposed new transport layer protocols to hide the intermittent characteristics of V2I WiFi communications and improve the throughput of the applications. Though, they require to update the network stack of the WiFi clients and deploy transport proxies in networks. While we would likely have observed better transport layer performance using their protocols in our experiments, it is impractical for today's applications to embrace them as both require network stack upgrades and deployment of proxies in the Internet. Since our goals are to understand how well today's urban WiFi networks can support vehicular applications as-is, we decided to use TCP as our transport protocol that is predominantly used by existing applications. *Hence, our system does not imply any modification to the existing network stack.*

Network Aided Approaches: Several studies [145, 146, 147, 148] proposed better AP handoff mechanisms to enhance the V2I WiFi performance. However, all of them require modifications to the APs and, therefore, control over the infrastructure. *In our work, we could not adopt any of their approaches as we have no control whatsoever over the infrastructure.*

Multipath TCP: Finally, Lee et al. [139] studied the usage of commercial city WiFi networks as one of the connectivity option for MPTCP [149] to enhance the performance of low-latency applications on connected vehicles. In their work, they gave an up-to-date view of the capability of today's in-situ WiFi networks from the network latency perspective. *Instead, our work focuses on the bandwidth perspective as we are targeting disruption-tolerant applications.*

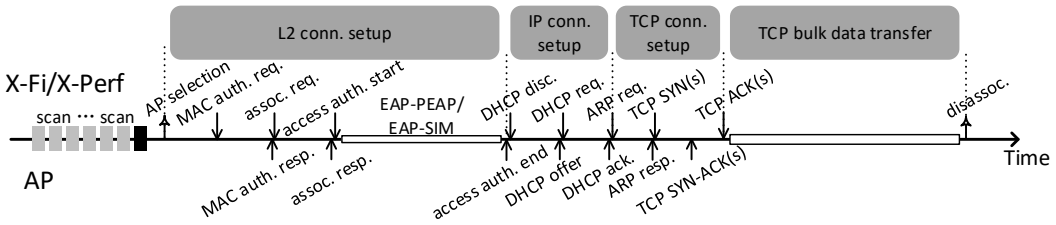

Fig. 8.1: Connection procedure handled by *X-Fi/X-Perf*.

8.3 System Overview

X-Fi/X-Perf is a ready-to-deploy toolkit for WiFi offloading in the wild. *X-Fi* integrates with the standard Linux WPA-Supplicant[5] and is designed to provide seamless access to commercial WiFi networks from a moving vehicle or in other high-mobility settings. *X-Perf* is a measurement tool to study application performance in intermittent connectivity scenarios. The flowchart in Figure 8.1 illustrates the various steps that are required to setup the connection.

8.3.1 *X-Fi*: connecting a moving vehicle to commercial WiFi hotspots

X-Fi has two main components: the vehicle agent, that runs in every *X-Fi*-enabled vehicle, and the orchestrator, that runs in the cloud. The **vehicle agent** consists of several different pieces: **(1)** a WPA and IEEE 802.1X supplicant, which includes a number of heuristics for access point selection, scanning, roaming, and layer 2 (L2) connection handling in general; **(2)** a lightweight DHCPv4 client customized for V2I; **(3)** a GPS receiver to keep geographical memory of the WiFi APs. The **orchestrator** maintains a database of APs crowd-sourced by *X-Fi*. For each APs, the database stores the GPS coordinates of the area and a rough estimate of the coverage and performance levels. The orchestrator database is meant to aid the AP selection. The vehicle agent can operate with or without the orchestrator support; more details can be found in the Appendix 8.9. Other AP selection approaches have been explored in Virgil [151], which needs to run an extensive set of measurements for each AP in range, which is unfeasible in the short time we have driving by.

X-Fi leverages a set of heuristics to accelerate the AP association process as well as the DHCP protocol exchange. For instance, various timeouts have been tuned throughout the whole association and IP address acquisition process, in order to minimize the time wasted in an idle state waiting for a reply from the AP, which may never arrive. In fact, access points that are too slow to respond get penalized during subsequent connection attempts, so that another, potentially faster, AP in

[5] `https://w1.fi/wpa_supplicant/`

range can be tried instead. Furthermore, *X-Fi* intelligently schedules its channel scanning periods, favoring more frequent and shorter scans instead of full scans of the entire frequency band. Active scans, where the client sends *Probe Request* frames to solicit a *Probe Response* from nearby APs, are also preferred to passive scans, where the client sits idle on each channel, listening for periodic AP beacons. Indeed, active scans are more effective in discovering responsive APs, can learn more information on the BSS, and take a substantially shorter time to complete. This allows *X-Fi* to take better roaming decisions in the future and reduces the amount of disruption to ongoing application traffic.

Scanning and AP selection The supplicant component internally maintains three main data structures: the BSS table, the BSSID blacklist, and a set of radio channels to be scanned at the next iteration. The BSS table stores information about neighboring BSSes discovered during past scans, most notably the BSSID, the channel frequency, the signal-to-noise ratio (SNR), and the timestamp of the last scan that detected this BSS. Entries are removed from this table after a configurable number of seconds, unless they have been refreshed by a more recent scan. The BSSID blacklist contains the BSSes that should be tried last during the next connection attempt. A BSSID is added to this list whenever the supplicant fails to associate with it due to either a timeout or an explicit rejection from the AP. This mechanism serves to temporarily reduce the priority of an AP, in order to give other APs a chance before retrying the failing AP again. The list is cleared after a connection attempt completes successfully.

High level, *X-Fi*'s connection procedure works as follows: **(1)** If no recent BSS information is available, a scan of the set of channels to be scanned is triggered. Initially, this set contains all the usable channels in the configured frequency bands (2.4GHz, 5GHz, or both), taking hardware and regulatory constraints into account. **(2)** Once the scan results are available, the BSS table is updated with the new information and the set of channels to be scanned is reset to its initial value. **(3)** The access point with the highest SNR in the BSS table is chosen, ignoring those that are in the BSSID blacklist, and a connection attempt is initiated. If it is successful, the blacklist is cleared and the procedure concludes. Otherwise, the BSSID is added to the blacklist and this step is repeated. **(4)** If there are no non-blacklisted BSSes left in the table, the BSSes in the blacklist are tried next. As before, this step is repeated until a connection attempt succeeds or all the blacklist entries have been tried and failed. **(5)** If the previous steps are still unsuccessful, all the channels that do not have any known BSSes are removed from the set of channels to be scanned. **(6)** The procedure restarts from step 1.

Sometimes, after a scan, *X-Fi* notices that one or more access points in range may be a better choice than the one it is currently associated with. In that case, an internal heuristic tries to estimate the throughput of the candidate AP based on the SNR and other BSS parameters (e.g., whether HT or VHT rates are supported).

Then, the supplicant makes the decision to roam to the new AP if its estimated throughput is higher than the current AP's throughput by a configurable amount (5 Mbps by default). This ensures that *X-Fi* always stays connected to one of the best APs available, while preventing excessive and wasteful roaming when the estimated performance difference between the APs in range is small.

IP address acquisition Once the supplicant has established L2 connectivity to the access point, it is necessary for the client to obtain an IP address in order to communicate with other hosts on the Internet. This is done through the DHCP.

This process can be a major bottleneck, as the DHCP server assigning the addresses is entirely outside our control and in some cases may be geographically distant from the AP location, thereby increasing the latency of the DHCP transaction. It is therefore essential to shorten this phase of the connection setup as much as possible through client-side optimizations. Unfortunately, all off-the-shelf DHCP client implementations that we tested turned out to be too slow, often needing several seconds to obtain an IP address. Thus, we decided to implement a custom DHCP client[6] optimized for V2I scenarios.

X-Fi's DHCP client features two main optimizations. First, the timeouts used in each phase of the protocol have been tuned and generally shortened, to more aggressively react to packet losses common in vehicular WiFi networks. Second, when the supplicant roams to a different AP with the same SSID, the DHCP client will optimistically assume that the previously assigned IP address is still valid and will skip the first phase of the protocol (DHCPDISCOVER), instead issuing a DHCPREQUEST to renew the previous address lease. Given that the lease time is typically in the order of minutes or hours, this optimization allows skipping a whole round-trip in the vast majority of cases.

8.3.2 *X-Perf*: measuring intermittent connectivity

There are several challenges to be overcome when evaluating uplink/downlink performances in commercial WiFi networks. In particular, it is necessary to cope with the intermittent nature of WiFi availability that causes unexpected interruption of TCP flows thus leading to inconsistent data. This is phenomenon can be easily observed when using well-known measurement tools as *iPerf* [152].

TCP underpins the vast majority of internet applications. Studying TCP performance over WiFi V2I scenarios is of paramount importance to identify the off-loading scenarios in the connected vehicle arena. *iPerf*'s lack of intermittent networks support[7] drove us in the design of *X-Perf* a dedicated performance measurement tool for intermittent networks. *X-Perf* monitors the status of the WiFi interface operated

[6]Most of the low-level protocol details of our client are based on the `sd-dhcp` code, part of the `systemd` project, `https://github.com/systemd/systemd/tree/v230/src/libsystemd-network`.

[7]`https://github.com/esnet/iperf/issues/835`

by *X-Fi*: once the network interface obtains an IP address, *X-Perf* starts to probe the performance of TCP over the wireless link. Specifically, it measures the TCP throughput by performing data bulk transfers between a remote server and the local on-board computer. As soon as the WiFi connectivity is interrupted, *X-Perf* stops the TCP throughput measurement and archives all logs and packet traces collected in this measurement. *X-Perf* repeats the above process each time *X-Fi* connects to an AP until the end of the experiment or until the vehicle is turned off. The *X-Fi*/*X-Perf* companion logs a wealth of information from L2 to the application including packet-level traces. The logs are geo-referenced and synced with the GPS. *X-Perf* can be programmed to run a specific experiment or to perform a random set of experiments. In our measurement privileged exploring the application design space using random TCP parameters for each APs. In particular, each time the vehicle is connected *X-Perf* randomly selects a target configuration from a pool of potential ones. Each configuration is a three-dimensional tuple identified by: **(1)** the direction of the data flow (whether we are testing upload or download); **(2)** the type of congestion control; (**3**) the number of concurrent TCP flows. In this work, we elect BBR [153] and CUBIC [154] as congestion control candidates. BBR is a recent work adopted by many Google services and well represents the latency driven rate-based congestion control class. TCP Cubic is window based and is the default Linux congestion control. It favours loss over latency as congestion signal. We measured the performance for both upload and download, and we select either 1, 2, 4, 8, or 16 concurrent TCP flows; however, the number of flows has a marginal impact on the performances. For each TCP test run in this work the parameters not listed in our configuration tuple (e.g. buffer size, etc.) are set by the operating system to the default values.

8.4 Measurement Methodology

We run our evaluation and data collections across four different cities in Europe (Bologna and Paris), in Asia (Macao) and in North America (Los Angeles). By choosing different locations in different countries, we evaluated our system in radically different urban scenarios, traffic patterns, and viability. Specifically: **Bologna** (Italy) is a good sample of a medium size, medieval city. It is characterized by narrow streets and, being a touristy city, has an unpredictable traffic pattern. **Macao** is, again, a medium size city. Located in Asia, it is characterized by a number of high-rise buildings as skyscrapers and casinos. **Paris** is an extremely touristy and crowded EU metropolis; it is characterized by a well-defined city center, around which revolve both tourism and business. Besides, the APs deployment strategy in Paris is peculiar. **Los Angeles** (LA) is a megalopolis; very populated, with an alternating of residential areas, working districts, and rural regions, LA presents an exceptionally regular traffic pattern.

Unlike in Macao, where we set up a permanent deployment on a vehicle running errands daily, in Bologna, LA, and Paris, we installed our equipment (§8.4.1) on

rental vehicles. In each city, we subscribed to one or more local providers offering access to urban WiFi APs. This gave us the WPA credentials/SIM cards needed to authenticate and connect while driving. Different cities, and different providers, have different authentication strategies: PEAPv0/EAP-MSCHAPv2 in Bologna, LA, and Macao; EAP-SIM in Paris. Besides, the access point deployment strategies we encountered during our data collection can be grouped into two main approaches: **(1)** leveraging existing hardware installed in customers' premises; **(2)** installing new dedicated APs. In the first approach, the providers rent the APs to their subscribers. The rented APs present two different SSIDs when in operation. While one is exclusively used by the customers and secured with WPA/WPA2-PSK, the other serves as a WiFi hotspot for public users. Therefore, both the physical capacity of the 802.11 channel and the available bandwidth of the connection, result to be shared between private customers and public users. Quality of Service (QoS) rules, such as token-bucket algorithms, may be applied on the APs to prioritize the Internet access of the customers. On the other hand, the second deployment strategy consists of the installation, done by the network providers, of dedicated APs at either indoor or outdoor spots.

To collect unbiased data, the drivers were agnostic to the network availability. Our goal is to evaluate *X-Fi* performance across different traffic patterns and various networking infrastructures; validate our research hypothesis, proving WiFi could be a well-grounded alternative to the cellular network for disruption tolerant vehicular applications.

8.4.1 Experimental Setup

The computer used for the data collection is a Slimpro SP675FP Fanless Mini PC. It features an Intel x64 3rd Gen Core processor, 3x3 MIMO full-size MiniPCIe 802.11n card (WLE350NX-7A), an u-Blox EVK-7N GPS unit, 16GB of RAM, and 300GB of SSD for storage; it runs Ubuntu 16.04.1 with 4.10.0-33-generic Linux kernel. While for the deployment in Macao we adopted three dipole antennas, in Bologna, Los Angeles, and Paris, we opted for three multi-polarised antennas. Using different antennas in Macao was not possible due to constraints imposed by the vehicle supplier; it is noteworthy that this limitation slightly affected the *X-Fi* performance in Macao scenario.

At the on-board computer boot, *systemd* launches *X-Perf* (§8.3.2). Once up, it executes *gpxlogger* and *X-Fi* (§8.3.1), respectively to track the GPS coordinates and to start the association and connection attempts with the WiFi hotspots. As soon as the network interface operated by *X-Fi* obtains an IP address, *X-Perf* starts the measurements as in §8.3.2.

8.4.2 Connectivity Metrics

Here now follows the list of metrics used to evaluate *X-Fi* and to understand the L2 and layer 3 (L3) characteristics of the V2I WiFi connectivity.

WiFi association time: the time elapsed from the moment in which the association process starts to the establishment of link-layer connectivity.

Time until the IP acquisition: the time elapsed from the beginning of the association to the acquisition of an IP address.

Link-layer connectivity duration: the time between the connection at link-layer and the disassociation.

IP connectivity duration: the period during which the WiFi interface of the on-board computer holds an IP address.

Link-layer connectivity hole: idle period between two consecutive associations.

IP connectivity hole: idle period between two consecutive associations with IP address.

Initial signal strength (ISS): the ISS of an association is the signal strength sensed during the WiFi scanning phase. It is the only metric considered by the stock WiFi implementation during the AP selection phase. In a stationary case, selecting the AP with the highest ISS works well. We investigate whether this works as well, for in-motion vehicles.

Average vehicle speed: For each association, we cross-refer the corresponding GPS data on its timestamp to compute the average vehicle speed over it.

IP address roaming support: analyzing the IP address at each association, we can observe how often a vehicle changes its IP address during a drive. This affects the design of both applications and transport protocols in the scenario where in-motion vehicles fetch/upload content via commercial WiFi networks. TCP-based vehicular applications have to carefully handle the IP handover, as the TCP connection would break otherwise.

8.4.3 Network Performance Metrics

We now present a list of the metrics used to assess the performance of the network in an intermittent connectivity scenario.

Time until the first TCP ACK: It is the time elapsed from when the association starts to the first TCP ACK received from the TCP server.

TCP connectivity duration: It is the time between the moment in which the TCP test starts and its termination. Since *X-Perf* stops the TCP tests right after the WiFi interface loses its IP address, the end time of the TCP connectivity coincides with the end time of the IP one.

TCP connectivity hole: The idle period between two consecutive associations having TCP connectivity.

Average TCP goodput: For each association in which we successfully launch a TCP test, we compute, from packet-level traces, the aggregated average goodput of the TCP connection.

Data volume: For each association in which we successfully launch a TCP test, we compute, from packet-level traces, the total number of bytes sent, or received, during the TCP connection. When uploading data, we discard the bytes not acknowledged by the server.

8.5 Data Collection

We configured *X-Fi* to work with nine different commercial WiFi networks during our data collection.

Permanent deployment in Macao

In Macao, a local institution granted us the permission to permanently instrument a van they provided. We placed a Slimpro SP675FP Fanless Mini PC under the back seats. Two WiFi antennas and the GPS antenna were installed on the roof, while, due to the limitation of the van, we had to put the third WiFi antenna under the back seats. The on-board computer is mainly powered by the van. To enable a graceful shutdown of the computer we adopted a small backup battery. The van runs errands on behalf of the local institution and transport staff. For the whole duration of the experiment[8], the mobility pattern of the van remained unaltered from its usual one, allowing us to collect *in-the-wild* samples representative of taxi drivers and transporters real-world mobility. We deployed the TCP server used for our tests on a virtual machine (VM) rented from a cloud datacenter in Singapore to reduce the harmful effects on TCP performance caused by long round-trip-time.

Short-term driving in Paris

In Paris, we instrumented a rental vehicle with our equipment: three WiFi antennas and one GPS antenna are placed on the roof of the vehicle, the computer (Slimpro SP675FP Fanless Mini PC) is located on the front seat and powered by the vehicle. A *Dekart* SIM card reader is used for the EAP-SIM authentication required by *Free* [155]. We deployed the TCP server on a machine located in Paris. We drove along randomly selected routes trying to maximize the exploration of the city.

Short-term driving in LA and Bologna

The rental vehicle used in LA was equipped in the very same way as the one in Paris, minus the SIM card reader. Here, we methodically mapped both residential and commercial areas. We deployed the TCP server used for our tests on a cloud VM in LA. In Bologna, we replicated the Los Angeles setup. Regarding the TCP tests, we connected to our TCP server located in Paris.

[8]Note that the deployment is still active at the time of the writing of this manuscript.

	Paris	Bologna	LA	Macao	Total
Driving time analyzed	11h18m08s	12h48m29s	32h38m06s	222h07m21s	278h52m04s
Driving distance analyzed (km)	144.71	197.12	707.91	3202.82	4252.56
Avg. driving speed (km/h)	12.80	15.39	21.69	14.41	N/A
# of distinct days analyzed	4	5	7	205	221
# of assoc. attempts	4525	5559	6746	16410	33240
# of link-layer assoc.	1812	1501	1856	4074	9243
# of 2.4GHz assoc.	1812	1218	479	1073	4582
# of 5GHz assoc.	0	283	1377	3001	4661
# of IP acquisitions.	790	1302	1583	3793	7468
# of TCP tests	664	1179	1490	3578	6911
# of distinct associated APs	1610	917	1385	706	4618
# of distinct 2.4GHz APs	1610	796	391	263	3060
# of distinct 5GHz APs	0	121	994	443	1558
# of WiFi operators	1	3	3	1	8
Total TCP download (GB)	0.7	7	33.1	73.9	114.7
Total TCP upload (GB)	1.3	8.7	13.8	95.7	119.5

Table 8.3: Dataset summary.

8.5.1 Dataset Summary

The salient features of our dataset are summarized in Table 8.3. The WiFi networks we connected to in Bologna and Paris (*Fastweb*, Emilia Romagna regional provider, *ALMAWIFI* and *Free*) present a higher AP density than the ones in LA and Macao (*Spectrum*, *eduroam*, *TWCPasspoint*, *CableWiFi* and *Companhia de Telecomunicações de Macau* (CTM)). This leads to, on an average, more associations per hour: 160, 117, 57, and 18 respectively for Paris, Bologna, LA, and Macao. Remarkably, we observed a lower ratio of IP acquisitions to associations in Paris than in the other cities (43.6% in Paris, 86.7% in Bologna, 85.3% in LA, and 93.1% in Macao). This much lower DHCP success rate is caused by failures in responding to the DHCP requests, even with good link-layer connectivity ($-40dBm$). We confirmed that by connecting to some of these APs which failed to respond to the DHCP requests at locations where their signal strength was decent, between $-40dBm$ and $-50dBm$. As we have no control nor inside knowledge on the network infrastructure used in Paris, we could not go further explaining the root cause of that. Although the ratios of IP acquisitions to link-layer associations are much different across the four cities, the ratios between TCP tests over IP acquisitions are similar. Therefore, in each of the analyzed cities, once a moving vehicle obtains an IP address it is very likely able to establish TCP connections.

Figure 8.2 shows the complementary CDF (CCDF) of the average driving speeds per association in every city. Given the diversity of APs density and traffic conditions across the four cities, the average driving speeds per association are also dissimilar. In Paris, Macao, as well as in the downtown area of Bologna, where the speed limits are stricter, most of the associations happened at speeds lower than $30km/h$. Instead, outside the center of Bologna, as in LA, we had the chance to connect with outdoor APs placed along the main roads, driving at higher speed (40–62.5km/h).

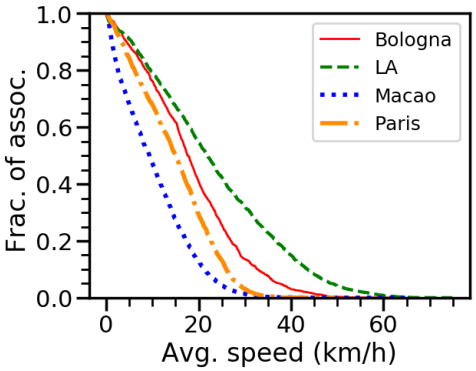

Fig. 8.2: CCDF of the average vehicle speed per association. Failed association attempts are not included. Mean: Bologna 18.42km/h, LA 23.43km/h, Macao 10.39km/h, Paris 14.83km/h.

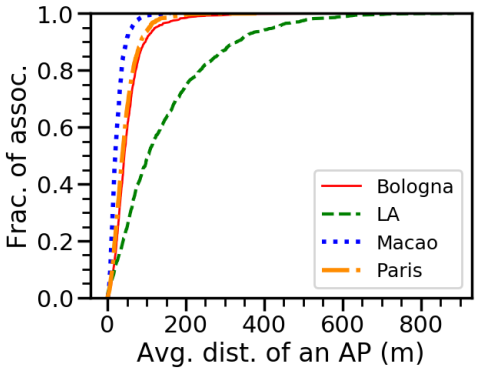

Fig. 8.3: CDF of the average AP coverage. Failed association attempts are not included. Mean: Bologna 53.40m, LA 145.54m, Macao 24.75m, Paris 45.05m.

Interestingly, from the data, we could not draw any evidence that lowering the driving speed leads to more successful associations. This denotes how the speed of vehicles does not have a direct effect on the association establishment success.

Figure 8.3 reports the CDF of the average coverage of each AP, computed based on the in-coverage distance of every association on the AP. The APs in LA present, on an average, a much wider coverage ($145.54m$) than the ones in the other cities, denoting how the providers placed several outdoor APs along roads and intersections.

8.6 Micro Benchmarks

This section presents the evaluation of *X-Fi*, capable of quickening the connection process, leading to longer intervals exploitable by disruption tolerant V2I applications. Our results confirm the efficiency and generality of *X-Fi* showing the impact of factors as speed, ISS, radio frequency.

8.6.1 WiFi association time

The CDFs of WiFi association time is reported in Figure 8.4. In Bologna, Macao and Paris, 90% of the associations terminate in less than one second. On the other hand, on an average, in LA the association process takes slightly longer than in the other cities. None of the distributions is heavy-tailed, with the means only lightly skewed from the medians. The data points with relatively longer association time are due to the re-transmission of the packets (i.e. EAP packets) necessary to set up the connectivity.

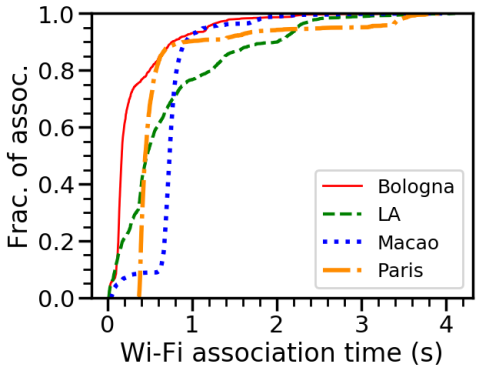

Fig. 8.4: CDF of the WiFi association time. Failed association attempts are not included. Mean: Bologna 0.34s, LA 0.73s, Macao 0.74s, Paris 0.67s.

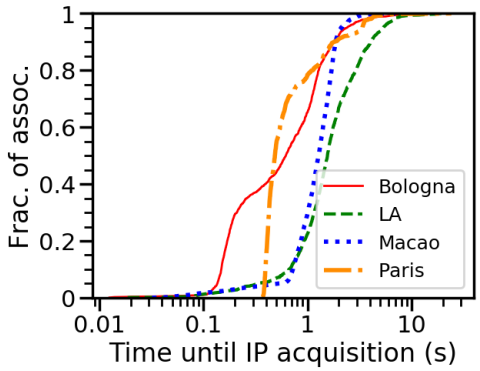

Fig. 8.5: CDF of the time until IP acquisition. Failed IP acquisition attempts are not included. Mean: Bologna 0.90s, LA 2.07s, Macao 1.28s, Paris 0.92s.

8.6.2 Time until IP address acquisition

The CDF of the cumulative time from the start of the association process to end of the DHCP process, is plotted in Figure 8.5. In Bologna, Macao, and Paris, the 90% of associations with IP connectivity is established within 2 seconds from the beginning of the association process. As the association time, also the DHCP process in LA is slightly longer than in the other cities. A more detailed breakdown of the phases prior gaining the IP (Figure 8.6) shows how EAP/PEAP authentication and DHCP account for most of the overhead. As expected, the DHCP and EAP authentication account for the major part of the overhead. This is because: 1) the DHCP and EAP authentication cause more packet exchange between clients and APs than the other two; 2) during the DHCP and EAP authentication phase, clients communicate with DHCP servers and authentication server normally in core networks, which causes longer latency than communication between clients and APs.

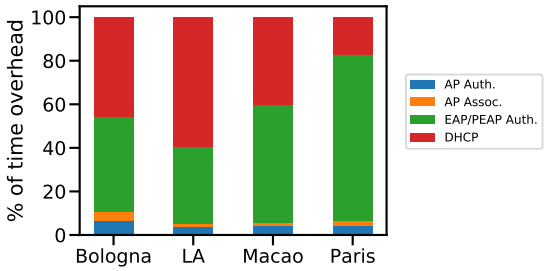

Fig. 8.6: Breakdown of the overhead before establishing the IP connectivity. All the values of each phase are averaged over all associations having IP connectivity. The average length of this overhead is reported in Figure 8.5.

8.6.3 Link-layer connectivity duration

Figure 8.7 shows the CCDF of the link-layer connectivity duration. Median values are 9.22s in Bologna, 17.29s in LA, 8.01s in Macao, and 9.04s in Paris. Noticeably, the distributions are heavy-tailed, with a few long-lasting connections due to traffic lights and traffic jams, typical of the urban mobility.

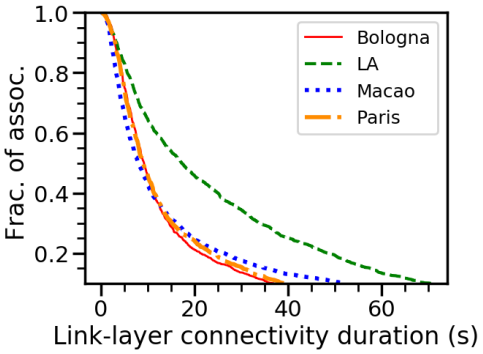

Fig. 8.7: CCDF of link-layer duration. Failed association attempts are not included. Mean values: Bologna 16.74s, LA 29.28s, Macao 21.75s Paris 16.58s.

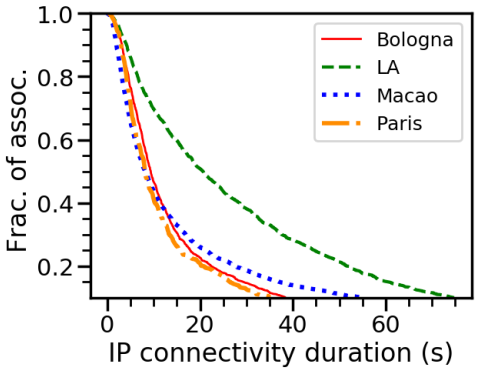

Fig. 8.8: CCDF of IP duration. Associations without IP are not included. Mean values: Bologna 17.47s, LA 31.91s, Macao 22.51s, Paris 15.84s.

8.6.4 IP connectivity duration

Figure 8.8 depicts the CCDF of the IP connectivity duration. Like for the link-layer, also these are heavy-tailed. For Bologna, LA, and Macao, both mean and median of IP connectivity duration are slightly longer than those at link-layer. This is due to the exclusion of some failed IP acquisitions[9], which are likely to have short link-layer duration.

8.6.5 Connectivity Holes

Due to the limited coverage of the existing urban WiFi deployments, there are gaps between consecutive associations. These reflect the intermittency of the connectivity the vehicles have to deal with if connected to commercial WiFi networks.

Link-layer connectivity holes

The CDF of the duration of the link-layer connectivity holes (Figure 8.9) reflects the frequency at which vehicles encounter APs, denoting their density in the region.

[9]The figures for the failed attempts are reported in Table 8.3.

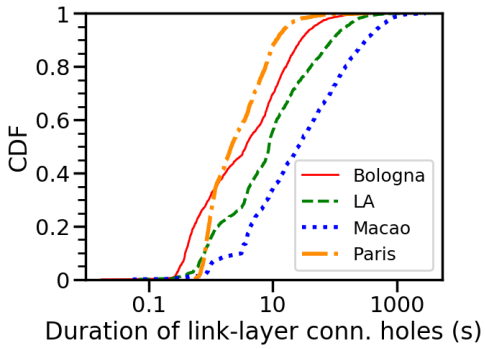

Fig. 8.9: CDF of the duration of the link-layer connectivity holes. Mean values: Bologna 12.23s, LA 33.08s, Macao 111.25s, Paris 5.70s.

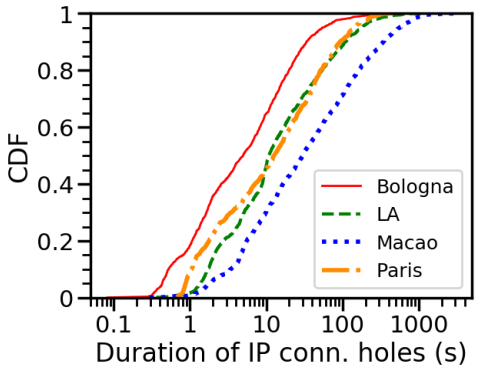

Fig. 8.10: CDF of the duration of the IP connectivity holes. Mean values: Bologna 15.69s, LA 40.70s, Macao 119.62s, Paris 34.42s.

Both in Bologna and in Paris, the AP density is capillary, with a median inter-arrival time between two consecutive associations of 3.69s and 2.13s, respectively. On the other hand, APs in LA and Macao are rarer, with longer median inter-arrival time (8.64s and 25.72s). Since, in Paris we only drove through densely populated neighborhoods, the mean is not too far from the median, with no sudden nor significant changes in the AP density across different locations. For all the other cities, the mean values are skewed from their median by a factor 4x, meaning the holes at the tail of the CDF are particularly long. This denotes how the AP distribution changes significantly across different regions in Bologna, LA, and Macao. The presence of a few extremely short holes (tens of milliseconds) is due to de-authentication packets sent by some APs to force the disassociation from them.

IP connectivity holes

As not every association succeeds in acquiring the IP address[9], we expect the duration of the IP connectivity holes to be enlarged by those associations which could not get the IP. For Bologna, LA and Macao, the mean values of the IP connectivity holes duration (Figure 8.10) are only marginally longer than those of the link-layer connectivity holes duration. This is because only a small fraction of associations in Bologna, LA, and Macao did not acquire the IP address. Though, in Paris, the mean duration of the IP connectivity holes is more protracted than its counterpart at the link-layer. This is a consequence of the DHCP issue discussed in §8.5.

8.6.6 IP Address Roaming Support

We investigated how the IP address assigned by a commercial WiFi network changes across different APs during a continuous drive. Usually, the IP address roaming

(the ability of a host to retain the same IP address when roaming across APs) is not supported when a WiFi client switches to another operator's WiFi network, unless the operators have a particular agreement. Hence, we focus on the IP address roaming inside WiFi networks operated by the same provider.

Specifically, we want to answer the following questions: **(1)** Can a vehicle keep its IP address unchanged during a continuous drive? **(2)** If not, for how long can a vehicle keep the same IP address?

Table 8.4 reports the data collected for every WiFi network[10] that we have used during our measurement campaign. The column IP duration reports the average time during which vehicles keep an IP address unchanged during a continuous drive. The IP duration values shown are formatted as *average* ± *standard deviation*.

City	Wifi network	IP changed?	IP duration (s)
Bologna	ALMAWIFI	✓	687 ± 1153
	Fastweb	✗	N/A
	EmiliaWifi	✓	184 ± 594
LA	Spectrum	✓	1644 ± 2653
	CableWifi	✓	3105 ± 5097
	Eduroam	✓	106 ± 156
Macao	CTM	✗	N/A
Paris	Free	✓	216 ± 384

Table 8.4: IP address roaming in different WiFi networks.

City	WiFi frequency	Link-layer duration (s)					#
		mean	stderr	median	20th %	80th %	of assoc.
Bologna	5GHz	26.22	39.62	11.70	5.91	36.49	283
	2.4GHz	14.54	19.56	8.58	4.31	18.51	1218
LA	5GHz	30.09	33.38	18.03	5.93	50.32	1377
	2.4GHz	26.94	30.67	15.26	5.38	41.28	479
Macao	5GHz	20.61	40.95	7.60	3.10	23.42	3001
	2.4GHz	24.96	42.73	9.28	3.87	32.90	1071

Table 8.5: Correlation between AP frequency and duration.

From our results, as only two networks apparently support it, we could not find any strong evidence of commercial WiFi networks supporting, by default, IP address roaming. Hence, either applications or transport layer protocols should take care of it as, otherwise, switching AP may break the connections established by connection-oriented protocol like TCP. For some of the legacy vehicle applications relying on TCP, Multipath TCP [149], which natively supports IP handover, could be a good replacement in order to leverage the intermittent connectivity provided by commercial WiFi networks without modifying the applications. However, the scope of this

[10]As we only get 12 associations from the TWCPasspoint WiFi network, we are not able to analyze it.

work is to assess whether current disruption tolerant applications could be carried out exploiting WiFi AP rather than cellular network, we focused our attention solely on TCP, as the most used transport layer protocol for current applications.

8.6.7 Impact of Speed, ISS, and frequency

Lastly, we studied which factors affect the V2I WiFi connectivity. Specifically, we wanted to analyze the impact of driving speed, ISS, and APs frequency on the V2I WiFi connectivity. Due to their similar nature, we only present the results for the link-layer connectivity.

Average vehicle speed

The first factor we consider is the driving speed. Intuitively, higher velocities cause shorter connectivity duration, as the vehicle leaves the AP coverage radius quicker. Figure 8.11 confirms that, correlating average speeds and link-layer connectivity duration.

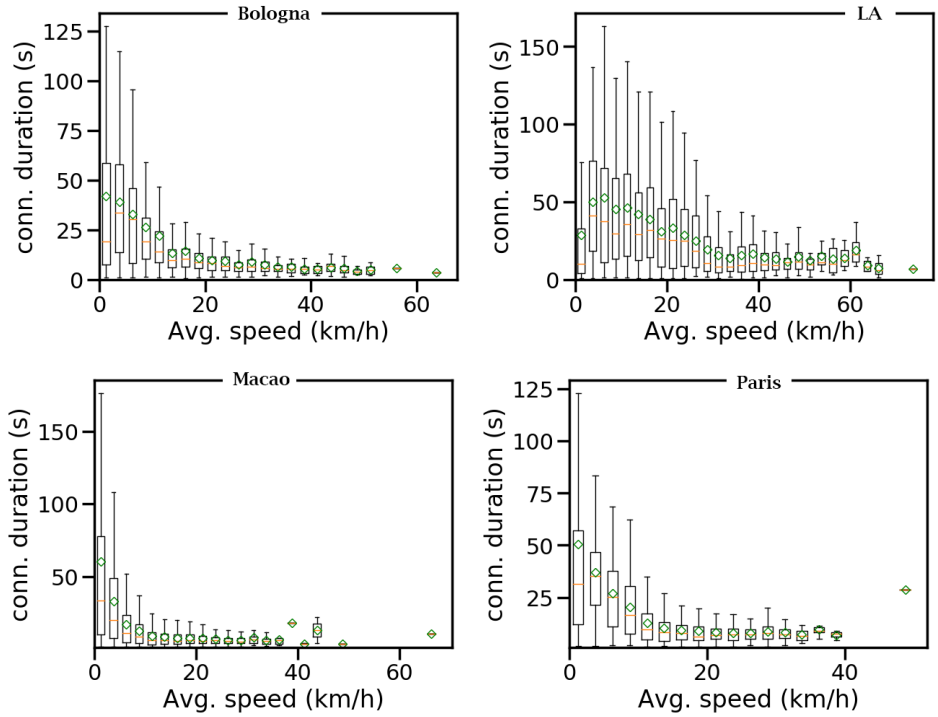

Fig. 8.11: Correlation between driving speed and connectivity duration: speeds are split into 2.5 km/h ranges. Each box represents a range which contains the associations with average speeds in that range. Red lines and green diamonds represent median and mean. As expected, slower velocities are likely to cause longer connections.

Initial Signal Strength

Being the only metric used during the AP selection phase by the stock WiFi implementations, we now consider the Initial Signal Strength (ISS). In the stationary case, choosing the AP with the highest ISS has proven to work well. However, due to the continuous changes of the clients position caused by the vehicular mobility, the ISS alone is not enough to infer the network performance.

In all the cities, the ISS ranges from being very weak ($-80dBm$) up to an excellent signal level (greater than $-50dBm$). To understand whether is enough to use the ISS as the only metric to infer, a priori, the network performance, we plot the correlation between the ISS and the link-layer connectivity duration (Figure 8.12). From the plots, we can not evince any obvious pattern confirming how associations with higher ISS last longer when mobility is involved. Instead, surprisingly, the duration of associations with higher ISS appears shorter than the ones with a lower one. This may be due to the vehicle moving from one extremity to the other of the AP coverage when it senses a weak ISS, while the vehicle could be already leaving the middle of the AP coverage when it senses a strong ISS. Hence, more sophisticated AP selection strategies, based on not only ISS, could improve the performance of vehicular connectivity.

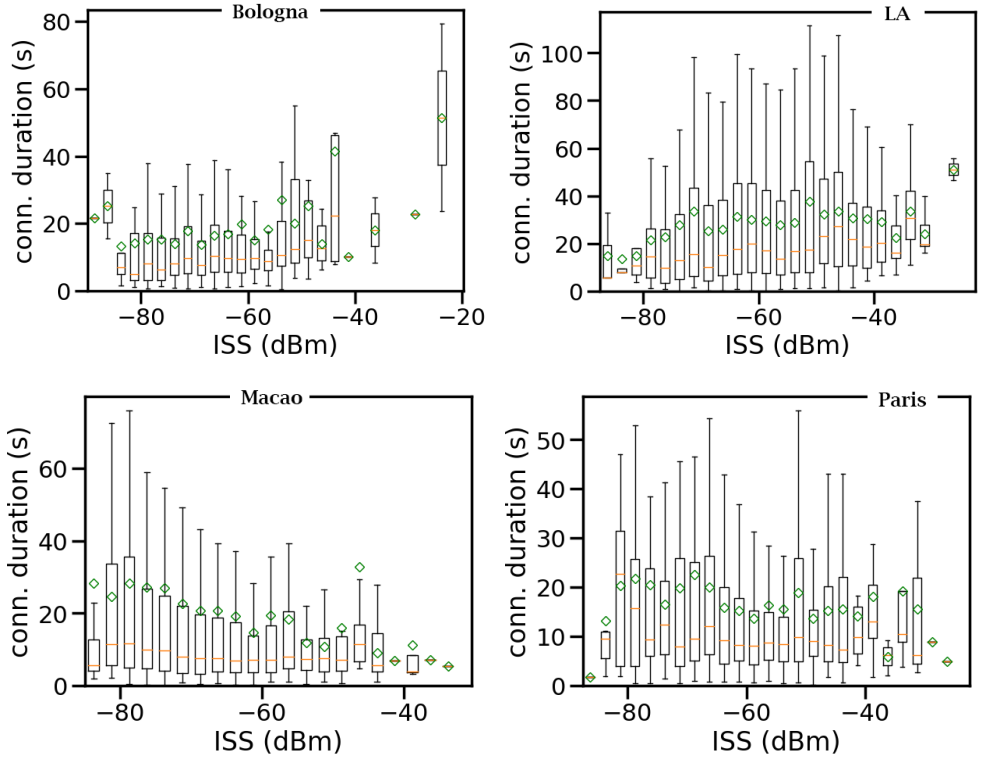

Fig. 8.12: Correlation between ISS and connectivity duration. We drew the box-whisker plot as in Figure 8.11. There are no obvious correlations between strong ISS and longer connectivity duration.

APs Operating Frequency

The last factor we take into account is the frequency of the WiFi APs. Since Free has only 2.4GHz APs available in Paris, we omit it. Table 8.5 reports the correlations between the APs frequency and the connectivity duration. Interestingly, from our results, the duration of the connectivity offered by 5GHz APs is usually better than the one offered by 2.4GHz APs. This is mostly due to a severe RF interference in 2.4GHz channels, causing bursty beacons losses, interrupting the connectivity.

8.7 Network Performance Analysis

The aim of this section is to prove WiFi as a well grounded alternative to cellular network for disruption tolerant V2I applications. We investigate: **(1)** the average goodput a vehicle can experience from an association; **(2)** the amount of data that can be transmitted during a single association. We do not distinguish between TCP tests with different parameters here. Although our results may not represent the best performance achievable by using the optimal TCP configuration, we believe they accurately reflect the truth as, in reality, applications use various TCP configurations.

8.7.1 Time until the first TCP ACK

Figure 8.13 presents the CDFs of the time between the start of the association process and the establishment of the TCP connections, i.e. the first TCP ACK. On average, the time that takes until the first TCP ACK is 1.09s in Bologna, 2.76s in LA, 1.46s in Macao, and 1.11s in Paris. Ultimately, the time to establish a TCP connection only accounts for a very small part of the entire association (between 10% to 18%, Figure 8.15), meaning that, even for short connections, for more than the 80% of the time that we are associated to an AP, it is possible to upload/download data.

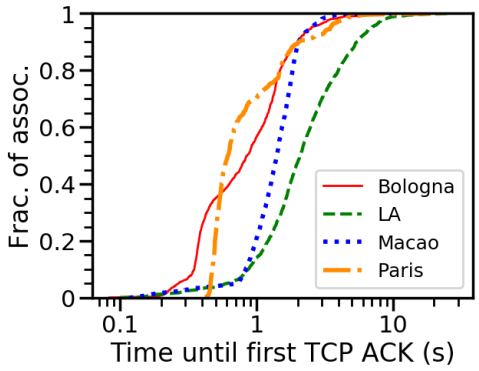

Fig. 8.13: CDF of the time until the first TCP ACK. All associations without being able to establish TCP connections are not included.

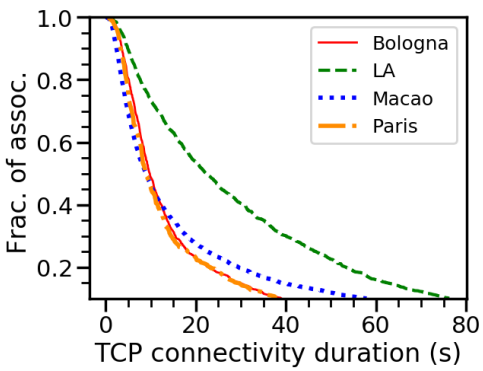

Fig. 8.14: CCDF of TCP connectivity duration. Associations not able to establish TCP connections are omitted. Mean: Bologna 18.08s, LA 33.57s, Macao 23.71s, Paris 17.24s.

8.7.2 TCP connectivity duration

The CCDF of the TCP connectivity duration is reported by Figure 8.14. The mean and median values of the TCP connectivity duration are slightly higher than the ones of the IP connectivity duration (§8.6.4) for the same reason mentioned in §8.6.4. The TCP connections last, on average, 18.08s in Bologna, 33.57s in LA, 23.71s in Macao, and 17.24s in Paris, opening the door to a multitude of WiFi-enabled disruption tolerant vehicular applications.

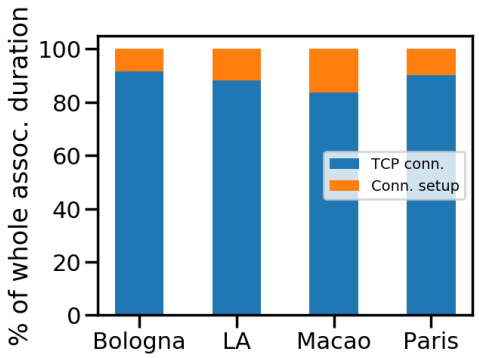

Fig. 8.15: Average ratio of TCP connection duration against the time needed to set it up. In orange, the TCP connection setup overhead.

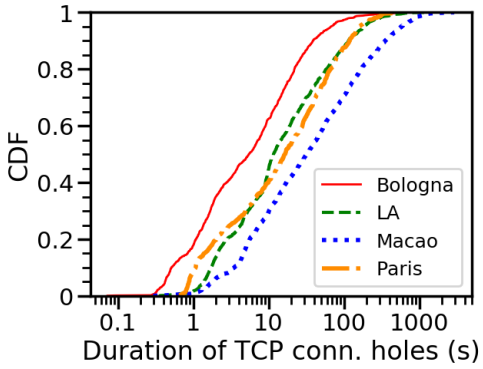

Fig. 8.16: CDF of the TCP connectivity holes duration. Mean: Bologna 18.13s, LA 43.45s, Macao 124.39s, Paris 42.45s.

8.7.3 TCP connectivity holes

From Figure 8.16 we can observe how the duration of the TCP connectivity holes is slightly longer if compared to the one of the gaps of IP connectivity (§8.6.5). This is because of the fraction of associations that failed to establish a TCP connection: 15.9% in Paris, 9.4% in Bologna, 5.9% in LA, and 5.6% in Macao. From our dataset, it is possible to conclude how, on any commute lasting more than a handful of minutes, vehicles are likely to encounter several usable APs.

8.7.4 Average TCP goodput

To support real world applications, a connection with decent goodput is fundamental. In Figure 8.17 we plot the CCDF of the average goodput per association, for both upload and download. The results in Macao seems very promising: for every association able to establish a TCP connection, the mean goodput is 13.19 Mbps in upload and 8.65 Mbps in download; with peaks up to 81.85Mbps and 90.48Mbps, respectively. Peak performances are likely to be achieved when vehicles, due to traffic jams or lights, stop at places where there are good APs. The results from LA and Bologna are also quite encouraging. On the contrary, the TCP performance in Paris is not as good, with the 90% of the associations providing less than 2Mbps for both upload and download. Like in the other cities, also the CCDF of Paris is heavy-tailed, biased by the vehicle stops by good APs. Notice, the maximum average goodput both in upload and download is ˜14Mbps.

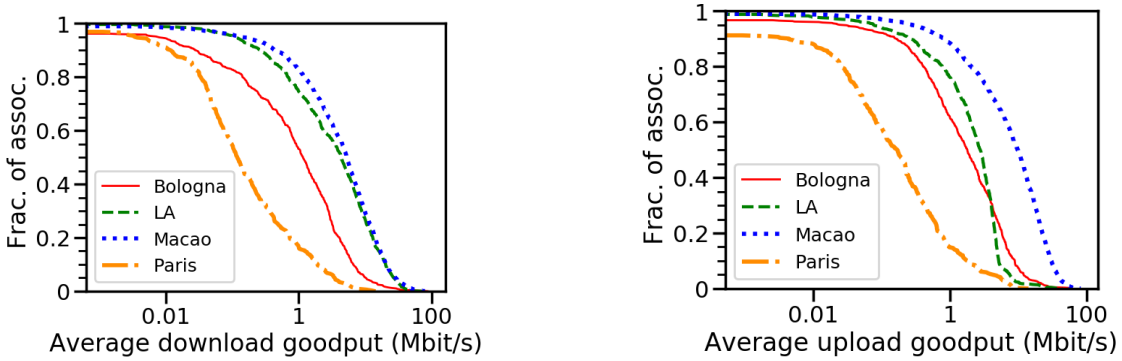

Fig. 8.17: CCDF of the TCP average goodput per association. All associations not able to establish TCP connections are excluded. Mean values (Download/Upload): Bologna 2.97Mbps/3.69Mbps, LA 7.62Mbps/3.21Mbps in LA, Macao 8.65Mbps/13.19Mbps, Paris 0.70Mbps/0.74Mbps.

8.7.5 Data volume

To have clearer insights on network capabilities in a similar scenario, we have to look at the actual amount of data uploaded/downloaded per association. Figure 8.18

presents the CCDF of the data volume transmitted during each association. Both in LA and Macao is possible to transmit a considerable amount of data during every association. More into detail, the mean data volume downloaded per association is 45.22MB and 42.23MB, up to 1.04GB and 2.26GB, respectively. Regarding the upload, on average, the amount of data uploaded per association is 19.15MB in LA and 54.63MB in Macao, with a maximum of 1.01GB and 1.86GB. Once again, in Paris, the results are less attractive, with only 3.88MB uploaded and 2.39MB downloaded on an average.These figures result skewed by the outliers at the tail of the distribution.

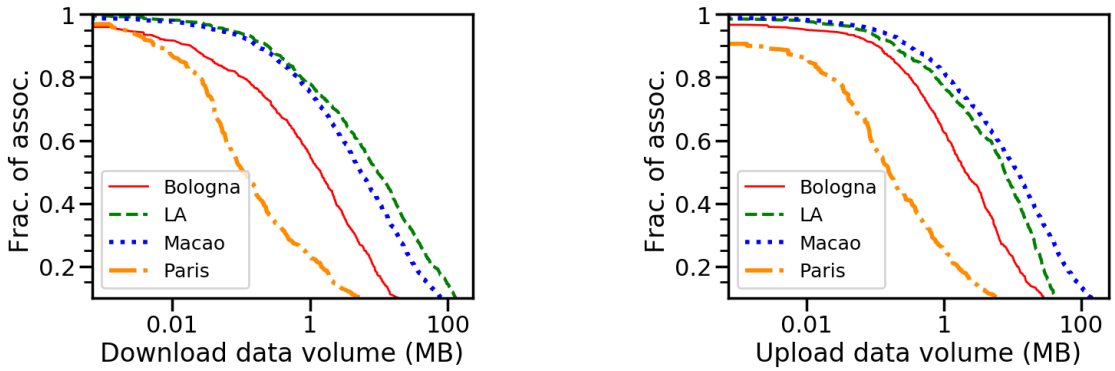

Fig. 8.18: CCDF of the data transfer size per association. All associations not able to establish a TCP connection are excluded. Mean transfer size (Download/Upload): Bologna 12.46MB/14.67MB, LA 45.22MB/19.15MB, Macao 42.23MB/54.63MB, Paris 2.39MB/3.88MB.

The reasons for the discrepancies between Paris and the other cities could be multi-folds. Firstly, the AP density in Paris is considerably higher than in the other cities and, therefore, we could suffer from higher RF interference in Paris. Secondly, only 2.4GHz APs are provided by Free. In general, 5GHz APs offer better performance, as they have higher modulation rate and less interfering channels. Lastly, the APs in Paris are shared between private customers and public users, with customers who are often prioritized by the providers. Besides, in the other cities, there are also dedicated APs, with better hardware and friendlier to public users.

Considering the total driving hours and the downloaded and uploaded data volume, our results from Bologna, LA, and Macao confirm that vehicles are likely able to download or upload around 1GB (1.23GB in Bologna, 1.44GB in LA, and 0.76GB in Macao) per hour by using stock TCP implementations, while on-the-go. Ultimately drawing important conclusions regarding the usage of commercial WiFi networks for disruption tolerant vehicular applications. Furthermore, this figure could improve with more APs coming into operation, and with a more capillary diffusion of 802.11ax APs.Moreover, as TCP is known to be inefficient when it comes to intermittent connectivity, this could get even better by using novel, alternative,

transport layer protocols.

8.8 Discussion and Final Remarks

In this work we presented the system we built and a characterization of V2I communication through WiFi offloads when using provider-managed networks, with authenticated access and encrypted data services. With over 278 driving-hours of measurements and 4252Km driven in four different metros in different continents, we tested the capabilities of *X-Fi* and assessed (*X-Perf*) the performance of today's networks for V2I applications. Our results show how our system handles the requirements a connected vehicle must manage nowadays: leveraging the existing WiFi infrastructures in an agnostic manner, without privileged access, *X-Fi* establishes an authenticated connection with encrypted commercial WiFi APs spread across the different cities. Our work proves it is feasible to establish authenticated connections, while on-the-go, exchanging content for tens of seconds per WiFi session.

X-Perf showed how exploiting provider-managed WiFi networks is a viable direction to manage the large amount of data produced by connected vehicles, reducing the automakers dependency from cellular constraints, and lifting them from the burden of the cellular network costs. Leveraging cheap WiFi subscriptions, *X-Fi* opens the door to a new kind of mobility where connectivity gains importance when choosing the daily commute routes. Downloading almost 1GB/h without additional costs for the driver, the vehicle will provide extra features such as updates, advanced support to the map navigator, virtual reality, or a renovated infotainment available to the driver and its family on board.

Ultimately, our work shows that, regardless the differences between the four cities we analyzed, *X-Fi* can enable disruption tolerant V2I applications exploiting urban commercial WiFi networks, lifting the automakers from the burden of cellular networks. Besides, reduces the load over the existing cellular infrastructure, slowing down the investment pace required to the providers.

8.9 *X-Fi* implementation details

8.9.1 *X-Fi* orchestrator role

The **orchestrator** maintains a database of APs crowd-sourced by *X-Fi* vehicles. For each AP, the database includes GPS coordinates of the area and a rough estimate of the coverage and performance levels. The orchestrator database is meant to aid the AP selection. Since APs often change channel and their performance can change at any time, the database is very dynamic and the vehicle agent must periodically request updates from the orchestrator.
Every time a vehicle successfully connects to an access point, it learns the performance characteristics of that network and, if possible, it submits the newly acquired

information back to the orchestrator, which then updates the AP database for future client requests. This is basically the same idea as crowd-sourcing.

When the vehicle agent has no connectivity, it operates in disconnected mode without orchestrator support. In this mode the vehicle agent scans and tries to connect to WiFi networks, but its roaming decision will solely be based on information available locally. Such for example RSSI and a few other parameters. In any case, the vehicle acquires and stores the data for all the APs it encounters once possible it will synchronize its local AP database with the database of the orchestrator.

8.10 *X-Perf* Implementation details

In details, *X-Perf* works as follows: immediately after the on-board computer boots, *systemd* launches *X-Perf*. Once this is up and running, it executes *gpxlogger* and *X-Fi*, respectively to track GPS coordinates and to start the attempts to associate with the WiFi hotspots. As soon as the network interface operated by *X-Fi* obtain an IP address, *X-Perf* starts the measurements.

In the interest of simplicity, we will refer to these two processes as *P1* and *P2*. P1 is responsible for capturing packet-level traces of TCP test using *tcpdump* [156]. P2, on the other hand, is in charge of performing the actual TCP test between a remote server and the local WiFi interface. As soon as the connectivity is interrupted and the WiFi interface, thereafter, loses the IP address, *X-Perf* automatically stops the execution of P1 and P2. In fact, this means a WiFi disassociation event has occurred, and, therefore, it is pointless to keep on sending/waiting for packets. Prior its actual termination, P2 ends its TCP test and saves the all the *.pcap* file created by P1, together with the profile of the TCP test. The latter includes the timestamps of the test starting time, the test ending time, and the configuration of the TCP test itself. In the meanwhile, *X-Perf* keeps on logging the GPS coordinates, monitoring the WiFi interface status, and measuring the TCP performance until the on-board computer shuts down, or a configured time limit is reached (this is further elaborated in Section 8.5).

Before the execution of *X-Perf* terminates, it kills all the running processes and compresses all the collected data and the log generated by *X-Fi* into a *.zip* package. If the time limit is reached, the tool immediately archives all data and restarts, ready to log the upcoming run.*X-Perf* can be programmed to run a specific experiment or to perform a random set of experiments.

Notice that given the brief duration of the connections, our setup is optimized to not to waste those RTTs needed to communicate the server the type configuration it should use for the forthcoming test. In fact, we instead open multiple sockets listening on different TCP ports at the server side, and we preset the configuration of each socket. This way, the server knows the configuration of any TCP test by its destination port number.

Chapter 9

C-Continuum

Processing high volumes of raw IoT data in the two different contexts mentioned in the previous chapters, poses a specific orchestration challenge. Although new IoT architectures can perform complex tasks directly on-device, lots of resources are still unexploited. Thus, the network edge naturally becomes a solution to share a common intelligence between the various nodes, even in highly dynamic networks, guaranteeing lower latency and higher performances compared to traditional computing offload towards the cloud.

9.1 Background

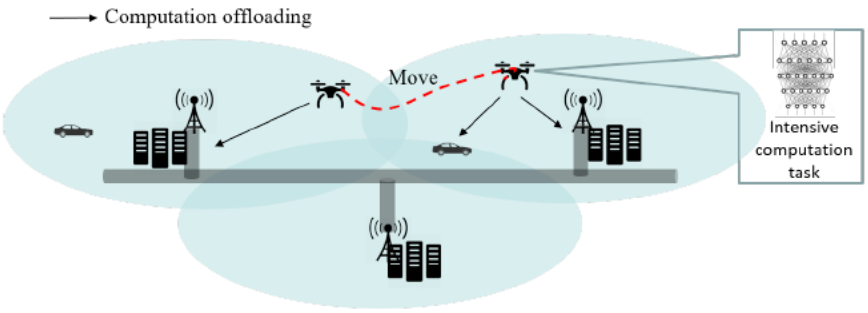

Fig. 9.1: Multi-Tier Computing across heterogeneous nodes

Mobile autonomous systems are supposed to deeply impact in manufacturing, space exploration, rescue, defense, transportation, and everyday life. Autonomous air-ground vehicles, for example, will become normal tools in the next few years, providing a natural platform for distributed artificial intelligence applications including, for example, disaster rescue and recovery, area surveying, autonomous driving, etc. The raise of autonomous cooperating robots will pose new challenges in networking,

distributed systems and resource management. Heavy computational tasks will be dispatched to the closest edge node for processing and the core-cloud will be involved as last resort in an effort to reduce latency and increase the global system capacity leveraging application and resource locality. Massive amounts of data and computations will be required. For example, in the autonomous driving scenario Intel estimates that each driver-less vehicle will produce over 4 TeraBytes of data each day[2]. While most of this data is consumed in-car, cooperating autonomous vehicles will have to exchange some percentage of the 4TB and eventually off-load some computation and data to the local edge or the core-cloud. This is particularly relevant when locally gathered and labeled data can be used to refine the model and ultimately increase the global *intelligence*. This approach is often taken by autonomous driving automakers.

Distributed AI applications demand effective, seamless, and efficient communication and computation mechanisms across the whole computing spectrum edge, fog, and cloud. In fig. 9.1, for example, computational tasks, AI models, as well as their inputs, parameters, and weights, may be instantiated on ground vehicles, in air-drones, and/or in the local edge data-center (DC). The allocation may and will change frequently during a single execution. Despite Machine Learning (ML) is now radically more efficiently applied to industry segments and data analysis, it still struggles to percolate into small devices. This becomes even harder if the small device is also in motion as, for instance, an autonomous vehicle.

Uncertainty is one of the most significant discriminant traits of today's society. When it boils down to the communication domain, uncertainty often translates into the absence of stability and reliability. Current industrial and academics trends denote a gradual, yet steady, transition towards an ubiquitous edge computing paradigm. Sudden changes in the computing load may be extremely harmful to CPUs and GPUs. The more we transition towards the edge, the more communications become intermittent and unreliable. Changes in both the network and the availability of the services may occur both locally and in the infrastructure. Nonetheless, cybersecurity threats are increasing[3]. These are only a few among the open-challenges that are coming along with the advent of pervasive edge devices. Fuelled by near deployment of 5G, the Research Community is putting a furious effort to tackle most of the aforementioned challenges. Nevertheless, to the best of our knowledge, the literature is not exhaustive regarding dynamic task and resource allocation approaches for mobile edge devices.

The diffusion of containerization [157, 158] in the recent years, along with the develop of lightweight unikernels[159], has opened the door to new and challenging computing-sharing strategies: high speed and large bandwidth connections, achievable via clever approaches as X-Fi chapter 8), allow a single node of the network, to rely on the edge capability instead of being limited by its local resources. This is true for any general-purpose tasks, and a common fashion in IoT applications. Yet, it becomes even more interesting whenever small devices cooperate towards a global AI intelligence.

Pushing AI to the edge, recently referred to as *edge AI*[160], is a new emerging way to offload AI models training, or part of it, to a neighbor. IoT applications with strict low-latency constraints, like autonomous driving, drones orchestration, and augmented reality, constantly need to locally retrain their AI models with quasi-real-time data collected by peripheral sensors, actuators, and cameras. The computational resources remain close to where input data are produced and consumed, thus reducing latency to milliseconds, and only considering the cloud as the very last resort.

Conversely, cloud-based AI APIs for ML engines have, on average, sub-second response times[161]. While these are enough for web-based AI functionalities as trends predictions, finance modelling, sales forecasting, and image processing, they are not enough for real-time operations. Secondly, AI intelligence in the cloud is bottle-necked by connectivity, given the engine is usually outsourced far from the consumer (i.e. the node requesting resources). Privacy concerns and security threats are also confined to a well-defined perimeter; since the edge deals with data that will never get to the core. Although there are several prior studies on edge computing [3, 162, 163] and the benefits of moving computation towards the edge, the literature lacks an orchestrator design that aims at achieving cloud-alike performance, providing AI inference models allocation for mobile devices, where reliability is a key factor and intermittent connectivity a hurdle.

Here, we outrageously propose *C-Continuum*, a Computing Continuum framework targeting distributed AI in the mobile arena. C-Continuum aims at defining a new generation of tools and mechanisms tailored to enable fine-granularity computation, coordination and mobility management across the mobile-computing spectrum from the edge to the core. In traditional IP networks, nodes communicate by sending IP packets to a specific destination addresses. Even though the IP networking model is, as of today, the de-facto worldwide standard for end-to-end communications, it does not fit the intermittent scenario. IP, in fact, was designed to create a communication network where packets identify communication endpoints only. On the other hand, C-Continuum embraces the Named Data Networking (NDN) paradigm making a case for naming any computational entity and using those names for resource location, data transfers, and computing functions as well [164], with minor additions to the legacy paradigm.

NDN

NDN [15] transforms the concept of host-centric network architecture (i.e. IP) to a data-centric network architecture and is one instance of a more general network research direction called Information-Centric Networking (ICN). It aims to generalize the IP's thin-waist, such that packets can name objects other than endpoints in a communication.

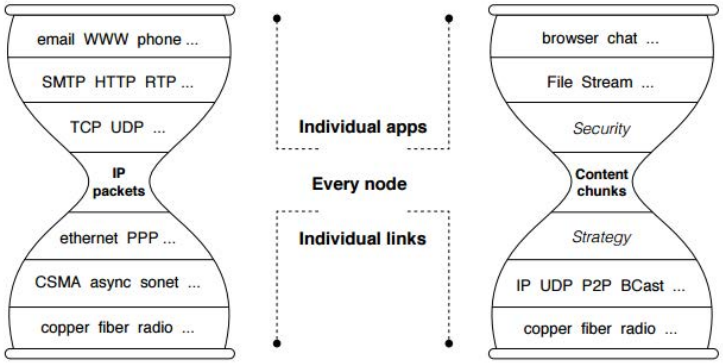

Fig. 9.2: Internet and NDN Hourglass Architectures [165]

NDN changes the semantics of network service from delivering packets to a given destination address to fetching data identified by a particular name. The name of an NDN packet could be the title of a book's chapter, a system-command, a specific sensor data, the frame of a video, etc. In our scenario, the name of an NDN packet can be an AI model, a computing resource, the weights of an Artificial Neural Network (ANN), services, or CPU cycles.

Communication in NDN is driven by the exchange of two different types of packets: (i) *Interests*, which contain the name of a desired authenticated content, and (ii) Data, registered locally, and then requested, that may be produced at one node in the network. All data are signed with the producer's key and there is an unique matching between an interest and its data.

In order to carry out the interest/data packet exchange, NDN routers are ruled by a Forwarding Strategy (FS), for selecting the execution location, with the help of three data structures:

1. Forwarding Information Base (FIB), is used to route interests; it is populated by a name prefix based routing protocol and multiple outgoing interfaces (*faces*), wired and wireless, to forward the interests to the contents. If the FIB receives an expired data packet, it cancels it.

2. Pending Interest Table (PIT), the table contains different records; every entry records the interests name with its incoming and outgoing related interfaces. Once the data has been discovered, consumed, or forwarded, the entry is cleared;

3. Content Store (CS), is a cache memory of data packets received for future reuse/re-transmission.

Interest packet processing

Whenever an interest packet comes into an NDN router, it looks first into the CS, at matching data; if it exists, the router sends the Data packet back on the correct interface, from which the interest previously came; otherwise, it looks up the name in its PIT. In the absence of a matching PIT entry, the NDN router forwards the interest towards the neighbors, based on the forwarding strategy stored in the FIB; otherwise, the PIT is updated with the new interest's face, and the request is discarded.

Data packet processing

Whenever a data packet is received, the NDN router first checks for a matching PIT entry, then it forwards the data to *all* the downstream interfaces listed by the PIT entry. Eventually, the router removes the PIT entry, caching the data in the Content Store. Every data packet follows the reversed path of the corresponding interest; if multiple interest packets were received before data production, the router duplicates the data on each downlink, ensuring the NDN symmetry (i.e. flow balance).

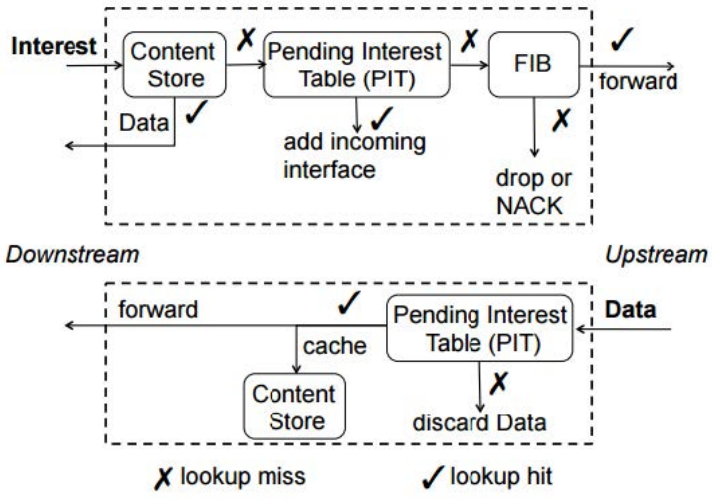

Fig. 9.3: Forwarding Process at an NDN Node [15]

Neither interest nor data packets carry any host or interface addresses. NDN routers forward interest packets towards data producers, based on the names carried by the packets, and send data packets to consumers based on the PIT state information set up by the interests at each hop. Hence, the need for any source or destination notion becomes superfluous.

9.2 System design

9.2.1 Overview

C-Continuum targets task and resource allocation in heterogeneous mobile scenarios, like autonomous vehicles, drones, and roadside micro data centers (μDC). In particular, C-Continuum:

1. proposes a framework with mechanisms, models, and algorithms to address complex domain-specific tasks such as collaborative mission-planning, computing, and networking issues in Air-Ground Vehicular Networks;

2. develops a system platform providing computing resources and AI models allocation;

3. smoothly integrates unmanned aerial vehicles (UAV), ground vehicles, edge nodes, and AI-based algorithms yielding more reliable communication among them.

Other edge-computing frameworks emerged in this domain during the last decade: NFN [166] was the first proposal of an NDN extension to support named functions in the network. Through the use of a λ-expression resolution engine, NFN's name identifies a content, a function, or a combination of the two. Similarly, Named Function as a Service (NFaaS) [167] exploits unikernels, lightweight virtual machines with available embedded functions stored in a dedicated table called *Kernel Store*. Recently IoT-NCN[162] formalized a new edge-computing framework for data retrieval from IoT sources; it selects an edge node as *service executor*, limiting the raw IoT data traffic across the network and, thus, allocating the execution based on the available nodes' processing resources.

C-Continuum is the first edge computing framework specifically designed to handle **intermittent networks**. Concretely, it is able to:

- request and assign tasks at any level, from in-vehicle specific tasks to data-center ones;

- offer computational resources and light-weight AI-models seamlessly addressable at, and by, any tier, independently.

Practically, the underlying idea is to label every C-Continuum node as a contributor that is in charge of running an unikernel configured by the consumer. For AI tasks, we leverage the Federated Learning (FL) [168] principles, where all the nodes work together to partially train a model based upon the demands of a requester, e.g. a vehicle is moving across the network.

When a C-Continuum node requests a task, it sends a 1-hop broadcast interest to the edge, looking for neighbors with sufficient computing resources (i.e. CPU cycles) to process and execute it. According to the edge-AI approach, the sender is

looking for nodes capable of mutualizing the model training. Whenever the sender receives the new model's weights/parameters from the edge, it can then proceed by internally merging them with the locally existing ones, thus resulting in a better performing ML model.

Unlike IoT-NCN, we exclude nodes but the one-hop ones since, otherwise, that would entail a larger processing interval between the request and the result. The exclusion of multi-hops nodes, along with the fast WiFi hotspot connectivity provided by X-Fi, pave the way to targeted task offload allocations. Moreover, it is not necessary anymore for every node to broadcast its own capabilities. Thus preventing the network from saturating. Instead, C-Continuum preliminary scans the edge-domain looking for an executor candidate, with a reactive discovery method, as described in §9.2.3.

The C-Continuum processing part is similar that of NFaaS. Once the fastest node acknowledges its availability to the consumer, (i) it sends a new interest requesting the function code and its input, encapsulated into an unikernel and, (ii) it executes the VM in a safe environment. In the meanwhile, the connection could drop, due to its intermittent nature. Thus, it is up to the consumer to restart the lost session, or to search for a new and more reliable executor (perhaps closer). In case of heavier computations, Long-lived-Interests [169] could be sent to maintain the pending requests longer in the PIT, ensuring the reception of the completed task via multi-hops routes at a later time. As a last resort, the consumer may select the cloud to offload the task.

9.2.2 Architecture

C-Continuum leverages NDN's typical consumer/producer behavior to offload and allocate tasks from the user to the local network and beyond. Thanks to NDN features such as multi-path forwarding, in-network caching, multicast data delivery, and data authentication, C-Continuum can allocate part of the workload to the closest capable node (i.e. the one satisfying the initial interest broadcasted by the user in the shortest time, and with enough resources to actually accomplish it).

The extensive reuse of caching and computation results minimizes the effectiveness of distributed denial of service attacks (DDoS)[170] against specific targets in the network. Yet, remote software execution comes along with other several major security concerns. Even if every NDN data packet is digitally signed with a signature forged as part of the data packet itself, the execution of malicious software may be fatal for the guest node. Thus, at any given time, the execution of any program must necessarily remain inside a sandbox.

9.2.3 Naming

Naming is the most challenging NDN component to take into account. The matching between a name and the data is unique in all the network. In our scenarios, thanks

to the flexibility provided by C-continuum, we can manage generic services from IoT to edge-AI. A semantically-rich naming is needed to identify all the possible resource categories which can be referred to with two strategies:

- resource-oriented: the consumer broadcasts multiple interests asking for enough memory (MB) and CPU cycles (GHz). This can be achieved by using the *ApplicationParameters* introduced with the NDN Packet v3. For instance, the sender can encapsulate /ndn/CC/cpu with the application parameter $cpu = 1GHz$, if at least 1GHz is the minimum requirement for the task; similarly, /ndn/CC/mem with $mem = 200MB$ indicates the RAM memory requested to store and execute the unikernel.

- category-oriented: instead of targeting numerical resources, C-Continuum categorizes the task contexts in pre-defined groups. For example, the consumer broadcasts /ndn/CC/IoT/aggregation or /ndn/CC/ML/classification whereas every node has already registered those names with customized local requirements.

We added /CC/ as a placeholder to address those applications which want to include C-Continuum as one of their possible strategies.

Once an executor becomes a candidate, it sends an interest back to the starting node, asking for an unikernel to run it. To do that, C-Continuum follows the *Non-routable transient name* pattern presented in [171]: our scenarios diverge from the typical *Pull* patterns where the client directly asks the producer for a content, which can either already be in its cache, or issued on-demand. The producer is a mere executor of a function that is initially available in the requester only.

Following this pattern, it is necessary to introduce an optional *DataLocator* field in the interest packet. The DataLocator contains meta-information, a token, that allows the edge node to retrieve the consumer's initial request. It exploits the PIT state as shown in fig. 9.4:

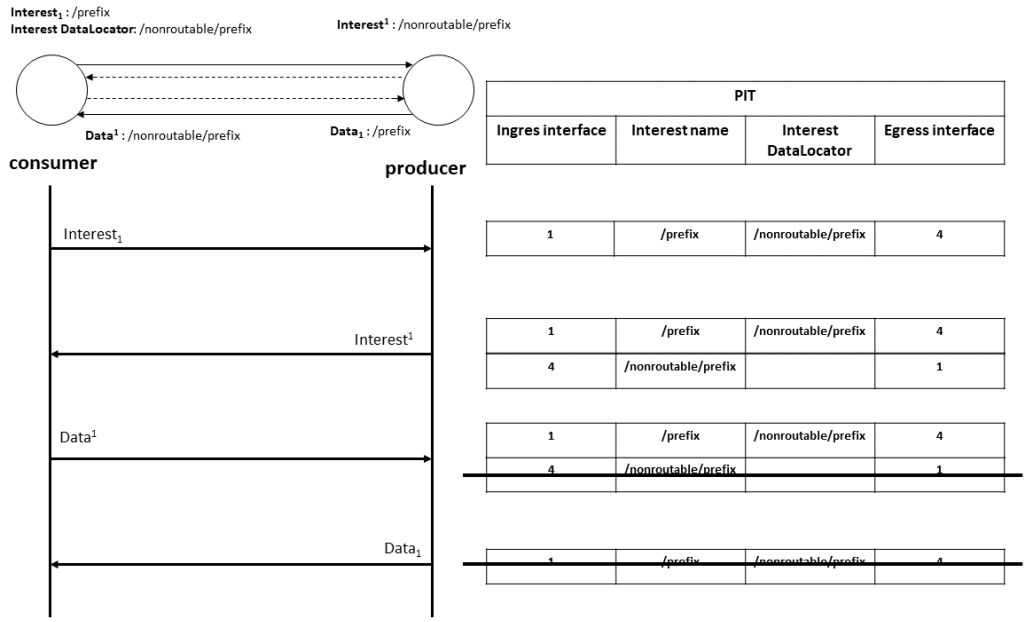

Fig. 9.4: C-Continuum consumer/producer workflow

1. The client constructs a non-routable prefix following the resource-oriented/category-oriented strategies described above; it also includes a unique token in the DataLocator field.

2. The interest arrives to the candidate node in the edge; its DataLocator token is saved in the PIT along with the name in the interest packet itself.

3. The producer verifies its free capabilities, according to the registered names, it starts a timer used for collision avoidance, and eventually replies with the task request interest using the non-routable token saved before. As the producer's interest is ready, NDN sends it to the correct face using the extended PIT entry. Thus, a new PIT entry is created for the non-routable name with the egress interface matching the ingress interface of the original interest; eventually, the request reaches the consumer, which can reply with the desired computing instance.

4. Finally, the task is carried out by the executor that replies with the completed task, i.e. the output.

9.2.4 Caching

In-network caching is one of the main characteristics of legacy NDN. Saving the contents at intermediate nodes minimizes the workload at the data producer side, and overall on the network. However, unlike the traditional IP caching approach, IoT

data typically expire soon and often have to be periodically refreshed by the producer. Moreover, the heterogeneous variety of tasks C-Continuum may orchestrate makes the caching not trivial.

There are two levels of caching that can be done by an NDN router: (i) short-term caching of packets provides for any re-transmission to overcome link congestion, client mobility, or lower-layer transport errors; conversely (ii) long-term caching of data contents is mainly for efficiently serving popular content. To deal with the intermittent scenario, C-Continuum caching policy is long-term. Although the resource requests are typically transient, C-Continuum can speed up the unikernel computations by caching them to avoid recompiling them every time; the output can also be retrieved again by sending the same interest along with the DataLocator token previously used.

9.2.5 Security

While TCP/IP delegates security responsibility to the endpoints, NDN secures the data itself by requiring data producers to cryptographically sign every data packet. [15] This inherently solves the lack of security in an IoT scenario. Further, it lead to fine-grained trust, allowing consumers to reason about whether a public key owner is an acceptable publisher for data.

Since NDN packets reference content rather than devices, it is harder to maliciously target a particular device, although mitigation mechanisms are needed against NDN-specific attacks, e.g. Interest Flooding DoS [170]. Task execution on the edge may represent a potential security vulnerability. Leveraging NFaaS signed unikernel approach, the program remains inside a sandbox where it runs without interfering with the external system.

Chapter 10

Conclusions

This thesis started with a couple of questions that followed the reader throughout the chapters: *Are there alternatives to the classical cloud-based approach? Is it possible to exploit the wealth of spare resources rather than directly offload the whole data to the server?*

This research aimed to identify a new IoT orchestrator that overcame some of the typical issues of the intermittent networks. The world of IoT is getting bigger and bigger, and this evolution is deeply impacting how we treat data, from the micro to the macro. Networking control, data sensing, optimization algorithms, resource allocation, distributed computing, and communication are only some of the IT branches that are keeping up with this unbridled growth.

The classic cloud approach was designed to manage a certain number of homogeneous cluster nodes; at present, these tools are incapable of tackling the complexity of a high degree of heterogeneity.

This work depicted how is easily feasible to generate millions of data, starting from simple weather stations on bikes up to deal with large amounts of fine-time granularity car data from an electric vehicle. The global IoT market size was valued at 14.9 Billion USD in 2019 and is projected to reach 58.9 Billion USD by 2026, with a Compound Annual Growth Rate (CAGR) of 21.7% between 2020 and 2026 [172]. Besides, the global autonomous vehicle market demand was estimated to be around 6.5 thousand units in the year 2019 and expected to witness a CAGR of 63.5% during the forecast period 2020 to 2027 [173]. This results in an exponential growth of new efficient and resource-demanding sensors and actuators that collaborate to local intelligence.

Even in our two scenarios, we saw how current computing platforms are underusing their capabilities and internal resources most of the time. On the other hand, data are generated at a greater pace than can be consumed by humans and the devices need to include efficient data orchestrators.

Leveraging X-Fi, a WiFi offloader for agnostic networks, and NDN, one of the promising next-Internet paradigm, this thesis tackled the issues aforementioned designing an edge-computing framework for tasks and AI models management, namely

C-Continuum. Through the straightforward process described in chapter 9, C-Continuum can provide heterogeneous services, such as:

1. V2I connectivity: e.g. via X-Fi itself, C-Continuum can enable multi-access network for vehicles, reducing by a 10 factor the time to access urban community WiFi hotspots;

2. an ICN fashion to distribute resources on the ground: e.g. using protocols like Navigó [174], an NDN approach to retrieve any name resource by connected cars;

3. AI modules for commodity tasks such as visual analytics: e.g. ParkMaster [175], a roadside parking-spot assistance service that proves how the *C-Continuum*'s edge-computing logic works in the context of Vehicle as a Service Provider.

Along with some architectural details and security concerns, the thesis eventually enhances the edge-computing panorama formalizing a new way to share functions and to overcome uncertainty in the mobile arena. Our results showed how this topic is still an open and challenging area, a long way from being exhaustively discussed in the literature.

References

[1] Paul Putland. "Connecting Devices: Access Networks". In: 2020, pp. 9–19. ISBN: 9781119545293. DOI: 10.1002/9781119545293.ch2.

[2] Intel. *For Self-Driving Cars, There's Big Meaning Behind One Big Number: 4 Terabytes*. 2017. URL: https://newsroom.intel.com/editorials/self-driving-cars-big-meaning-behind-one-number-4-terabytes/.

[3] Maurantonio Caprolu, Roberto Pietro, Flavio Lombardi, and Simone Raponi. "Edge Computing Perspectives: Architectures, Technologies, and Open Security Issues". In: 2019, pp. 116–123. DOI: 10.1109/EDGE.2019.00035.

[4] Erik Nygren, Ramesh K Sitaraman, and Jennifer Sun. "The Akamai Network: A Platform for High-Performance Internet Applications". In: *SIGOPS Oper. Syst. Rev.* 44.3 (2010), pp. 2–19. ISSN: 0163-5980. DOI: 10.1145/1842733.1842736. URL: https://doi.org/10.1145/1842733.1842736.

[5] Mahadev Satyanarayanan. "Pervasive computing: Vision and challenges". In: *IEEE Personal communications* 8.4 (2001), pp. 10–17.

[6] Dejan S Milojicic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja, Jim Pruyne, Bruno Richard, Sami Rollins, and Zhichen Xu. *Peer-to-peer computing*. 2002.

[7] Lizhe Wang, Gregor Von Laszewski, Andrew Younge, Xi He, Marcel Kunze, Jie Tao, and Cheng Fu. "Cloud computing: a perspective study". In: *New Generation Computing* 28.2 (2010), pp. 137–146.

[8] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. "The case for vm-based cloudlets in mobile computing". In: *IEEE pervasive Computing* 8.4 (2009), pp. 14–23.

[9] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. "Fog computing and its role in the internet of things". In: *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. 2012, pp. 13–16.

[10] Salman Taherizadeh Mohammad Hossein Zoualfaghari Simon Beddus. "Edge Computing". In: 2020, pp. 21–35. ISBN: 9781119545293. DOI: 10.1002/9781119545293.ch3.

[11] Larry Peterson, Tom Anderson, Sachin Katti, Nick McKeown, Guru Parulkar, Jennifer Rexford, Mahadev Satyanarayanan, Oguz Sunay, and Amin Vahdat. "Democratizing the Network Edge". In: 49.2 (2019), pp. 31–36. ISSN: 0146-4833. DOI: 10.1145/3336937.3336942. URL: https://doi.org/10.1145/3336937.3336942.

[12] Nikos Makris, Virgilios Passas, Thanasis Korakis, and Leandros Tassiulas. "Employing MEC in the Cloud-RAN: An Experimental Analysis". In: *Proceedings of the 2018 on Technologies for the Wireless Edge Workshop*. WirelessEdge '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 15–19. ISBN: 9781450359313. DOI: 10.1145/3266276.3266281. URL: https://doi.org/10.1145/3266276.3266281.

[13] Saro Velrajan. *Far Edge vs. Near Edge in Edge Computing*. 2019. URL: https://www.thetech.in/2019/06/far-edge-vs-near-edge-in-edge-computing.html.

[14] Rita Tse and Giovanni Pau. "Enabling Street-Level Pollution and Exposure Measures: A Human-Centric Approach". In: *Proceedings of the 6th ACM International Workshop on Pervasive Wireless Healthcare*. MobiHealth '16. Paderborn, Germany: Association for Computing Machinery, 2016, pp. 1–4. ISBN: 9781450343428. DOI: 10.1145/2944921.2944925. URL: https://doi.org/10.1145/2944921.2944925.

[15] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, K C Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. "Named data networking". In: *ACM SIGCOMM Computer Communication Review* 44.3 (2014), pp. 66–73.

[16] S Mirri, C Prandi, P Salomoni, F Callegati, A Melis, and M Prandini. "A Service-Oriented Approach to Crowdsensing for Accessible Smart Mobility Scenarios". In: *Mob. Inf. Syst.* 2016 (2016), 2821680:1–2821680:14.

[17] A Melis, S Mirri, C Prandi, M Prandini, P Salomoni, and F Callegati. "Integrating Personalized and Accessible Itineraries in MaaS Ecosystems Through Microservices". In: *Mobile Networks and Applications* 23 (2018), pp. 167–176.

[18] Nicola Dusi, Ilaria Ferretti, and M Furini. "PlayTheCityRE: A visual storytelling system that transforms recorded film memories into visual history". In: *2016 IEEE Symposium on Computers and Communication (ISCC)* (2016), pp. 85–90.

[19] Hafedh Chourabi, Taewoo Nam, S Walker, José Ramón Gil-García, S Mellouli, K Nahon, Theresa A Pardo, and Hans Jochen Scholl. "Understanding Smart Cities: An Integrative Framework". In: *2012 45th Hawaii International Conference on System Sciences* (2012), pp. 2289–2297.

[20] S Mirri, C Prandi, and P Salomoni. "Personalizing Pedestrian Accessible way-finding with mPASS". In: *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)* (2016), pp. 1119–1124.

[21] F Callegati, G Delnevo, A Melis, S Mirri, M Prandini, and P Salomoni. "I want to ride my bicycle: A microservice-based use case for a MaaS architecture". In: *2017 IEEE Symposium on Computers and Communications (ISCC)* (2017), pp. 18–22.

[22] Rita Tse, L Zhang, Philip Lei, and G Pau. "Social Network Based Crowd Sensing for Intelligent Transportation and Climate Applications". In: *Mobile Networks and Applications* 23 (2018), pp. 177–183.

[23] P Lena, S Mirri, C Prandi, P Salomoni, and G Delnevo. "In-vehicle Human Machine Interface: An Approach to Enhance Eco-Driving Behaviors". In: *SmartObject '17*. 2017.

[24] A Bigazzi and M Figliozzi. "Roadway determinants of bicyclist exposure to volatile organic compounds and carbon monoxide". In: *Transportation Research Part D-transport and Environment* 41 (2015), pp. 13–23.

[25] Óscar Alvear, W Zamora, C Calafate, J Cano, and P Manzoni. "EcoSensor: Monitoring environmental pollution using mobile sensors". In: *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)* (2016), pp. 1–6.

[26] W Lerner and Van Audenhove. "The future of Urban Mobility: Towards Networked, Multimodal Cities in 2050". In: *Public transport international* (2012).

[27] G Marfia, G Pau, E Giordano, E De Sena, and M Geria. "VANET: On Mobility Scenarios and Urban Infrastructure. A Case Study". In: *2007 Mobile Networking for Vehicular Environments* (2007), pp. 31–36.

[28] M Roccetti, M Gerla, C Palazzi, S Ferretti, and G Pau. "First Responders' Crystal Ball: How to Scry the Emergency from a Remote Vehicle". In: *2007 IEEE International Performance, Computing, and Communications Conference* (2007), pp. 556–561.

[29] V Ghini, P Salomoni, and G Pau. "Always-best-served music distribution for nomadic users over heterogeneous networks". In: *IEEE Communications Magazine* 43 (2005), pp. 69–74.

[30] S Mirri, C Prandi, P Salomoni, and L Monti. "Social Location Awareness: A Prototype of Altruistic IoT". In: *2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS)* (2016), pp. 1–5.

[31] F Callegati, W Cerroni, Chiara Contoli, and G Santandrea. "SDN Controller Design for Dynamic Chaining of Virtual Network Functions". In: *2015 Fourth European Workshop on Software Defined Networks* (2015), pp. 25–30.

[32] F Callegati, W Cerroni, Chiara Contoli, and G Santandrea. "Performance of multi-tenant virtual networks in OpenStack-based cloud infrastructures". In: *2014 IEEE Globecom Workshops (GC Wkshps)* (2014), pp. 81–85.

[33] F Callegati, W Cerroni, Chiara Contoli, and G Santandrea. "Implementing dynamic chaining of Virtual Network Functions in OpenStack platform". In: *2015 17th International Conference on Transparent Optical Networks (ICTON)* (2015), pp. 1–4.

[34] P Salomoni, C Prandi, M Roccetti, V Nisi, and N Jardim Nunes. "Crowdsourcing Urban Accessibility: Some Preliminary Experiences with Results". In: *Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter*. CHItaly 2015. New York, NY, USA: Association for Computing Machinery, 2015, pp. 130–133. ISBN: 9781450336840. DOI: 10.1145/2808435.2808443. URL: https://doi.org/10.1145/2808435.2808443.

[35] World Health Organization. *Urban health: major opportunities for improving global health outcomes, despite persistent health inequities*. 2016. URL: https://www.who.int/news/item/31-03-2016-urban-health-major-opportunities-for-improving-global-health-outcomes-despite-persistent-health-inequities.

[36] Judith C Chow, John G Watson, Douglas H Lowenthal, Paul A Solomon, Karen L Magliano, Stephen D Ziman, and L Willard Richards. "PM10 and PM2.5 Compositions in California's San Joaquin Valley". In: *Aerosol Science and Technology* 18.2 (1993), pp. 105–128. DOI: 10.1080/02786829308959588.

[37] X Querol, A Alastuey, S Rodríguez, F Plana, C Ruiz, N Cots, G Massagué, and O Puig. "PM10 and PM2.5 source apportionment in the Barcelona Metropolitan area, Catalonia, Spain". In: *Atmospheric Environment* 35 (2001), pp. 6407–6419.

[38] EPA. *Particulate Matter (PM) Basics*. 2016. URL: https://www.epa.gov/pm-pollution/particulate-matter-pm-basics.

[39] Luigi Atzori, Antonio Iera, and Giacomo Morabito. "The Internet of Things: A survey". In: *Computer Networks* 54.15 (2010), pp. 2787–2805. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2010.05.010. URL: http://www.sciencedirect.com/science/article/pii/S1389128610001568.

[40] Emily G Snyder, Timothy H Watkins, Paul A Solomon, Eben D Thoma, Ronald W Williams, Gayle S W Hagler, David Shelow, David A Hindin, Vasu J Kilaru, and Peter W Preuss. *The changing paradigm of air pollution monitoring*. 2013.

[41] Christian; Kramar Hans; Kalasek Robert; Pichler-Milanovic Nataša; Meijers Evert Giffinger Rudolf; Fertner. *Smart Cities - Ranking of European medium-sized cities*. 2007. URL: http://curis.ku.dk/ws/files/37640170/smart_cities_final_report.pdf.

[42] Victoria Adams, Sudeeksha Murari, and Christopher Round. "Biking and the Connected City". In: 2017, pp. 307–321. ISBN: 978-3-319-51601-1. DOI: 10.1007/978-3-319-51602-8_18.

[43] R White, I Paprotny, Frederick Doering, W Cascio, P Solomon, and L Gundel. "Sensors and 'apps' for community-based: Atmospheric monitoring". In: 2012.

[44] M I Mead, O Popoola, G Stewart, P Landshoff, M Calleja, M Hayes, José J Baldoví, M W McLeod, T Hodgson, J Dicks, Alastair C Lewis, J Cohen, R Baron, J Saffell, and R Jones. "The use of electrochemical sensors for monitoring urban air quality in low-cost, high-density networks". In: *Atmospheric Environment* 70 (2013), pp. 186–203.

[45] S Zampolli, I Elmi, F Ahmed, M Passini, G C Cardinali, S Nicoletti, and L Dori. "An electronic nose based on solid state sensor arrays for low-cost indoor air quality monitoring applications". In: *Sensors and Actuators B-chemical* 101 (2004), pp. 39–46.

[46] T V Tran, Nam Trung Dang, and W Chung. "Battery-free smart-sensor system for real-time indoor air quality monitoring". In: *Sensors and Actuators B-chemical* 248 (2017), pp. 930–939.

[47] Murat Dener and Cevat Bostancıoğlu. "Smart Technologies with Wireless Sensor Networks". In: *Procedia - Social and Behavioral Sciences* 195 (2015), pp. 1915–1921.

[48] S Moltchanov, I Levy, Y Etzion, U Lerner, D Broday, and B Fishbain. "On the feasibility of measuring urban air pollution by wireless distributed sensor networks." In: *The Science of the total environment* 502 (2015), pp. 537–547.

[49] A Al-Ali, I Zualkernan, and F Aloul. "A Mobile GPRS-Sensors Array for Air Pollution Monitoring". In: *IEEE Sensors Journal* 10 (2010), pp. 1666–1671.

[50] David Hasenfratz, O Saukh, Silvan Sturzenegger, and L Thiele. "Participatory Air Pollution Monitoring Using Smartphones". In: 2012.

[51] K Hu, Y Wang, A Rahman, and V Sivaraman. "Personalising pollution exposure estimates using wearable activity sensors". In: *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)* (2014), pp. 1–6.

[52] C Prandi, S Ferretti, S Mirri, and P Salomoni. "Trustworthiness in crowd-sensed and sourced georeferenced data". In: *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)* (2015), pp. 402–407.

[53] Wesley Willett, Paul M Aoki, N Kumar, S Subramanian, and Allison Woodruff. "Common Sense Community: Scaffolding Mobile Sensing and Analysis for Novice Users". In: *Pervasive*. 2010.

[54] E Bales, N Nikzad, N Quick, Celal Ziftci, K Patrick, and W Griswold. "Citisense: Mobile air quality sensing for individuals and communities Design and deployment of the Citisense mobile air-quality system". In: *Pervasive 2012*. 2012.

[55] S Kuznetsov and E Paulos. "Participatory sensing in public spaces: activating urban surfaces with sensor probes". In: *Conference on Designing Interactive Systems*. 2010.

[56] Karl Aberer, Saket Sathe, Dipanjan Chakraborty, Alcherio Martinoli, Guillermo Barrenetxea, Boi Faltings, and Lothar Thiele. "OpenSense: Open Community Driven Sensing of Environment". In: *Proceedings of the ACM SIGSPATIAL International Workshop on GeoStreaming*. IWGS '10. New York, NY, USA: Association for Computing Machinery, 2010, pp. 39–42. ISBN: 9781450304313. DOI: 10.1145/1878500.1878509. URL: https://doi.org/10.1145/1878500.1878509.

[57] C Prandi, M Roccetti, P Salomoni, Valentina Nisi, and N Nunes. "Fighting exclusion: a multimedia mobile app with zombies and maps as a medium for civic engagement and design". In: *Multimedia Tools and Applications* 76 (2016), pp. 4951–4979.

[58] I Martí, Luis E Rodríguez, Mauricia Benedito, S Trilles, A Beltrán, L Díaz, and J Huerta. "Mobile Application for Noise Pollution Monitoring through Gamification Techniques". In: *ICEC*. 2012.

[59] Z A Barakeh, P Breuil, Nathalie Redon, C Pijolat, Nadine Locoge, and Jean-Paul Viricelle. "Development of a normalized multi-sensors system for low cost on-line atmospheric pollution detection". In: *Sensors and Actuators B-chemical* 241 (2017), pp. 1235–1243.

[60] J Pereira, O Postolache, and P Girão. "Using neural network techniques in environmental sensing and measurement systems to compensate for the effects of influence quantities". In: *IEEE Instrumentation & Measurement Magazine* 17 (2014), pp. 26–56.

[61] Gaurav Pandey, Bin Zhang, and Le Jian. "Predicting submicron air pollution indicators: a machine learning approach." In: *Environmental science. Processes & impacts* 15 5 (2013), pp. 996–1005.

[62] X Liu, B Li, Aimin Jiang, Shixin Qi, Chaosheng Xiang, and N Xu. "A bicycle-borne sensor for monitoring air pollution near roadways". In: *2015 IEEE International Conference on Consumer Electronics - Taiwan* (2015), pp. 166–167.

[63] E Kanjo, J Bacon, D Roberts, and P Landshoff. "MobSens: Making Smart Phones Smarter". In: *IEEE Pervasive Computing* 8 (2009).

[64] C E Palazzi, S Ferretti, S Cacciaguerra, and M Roccetti. "On maintaining interactivity in event delivery synchronization for mirrored game architectures". In: *IEEE Global Telecommunications Conference Workshops, 2004. GlobeCom Workshops 2004.* (2004), pp. 157–165.

[65] S Mirri, C Prandi, P Salomoni, F Callegati, and A Campi. "On Combining Crowdsourcing, Sensing and Open Data for an Accessible Smart City". In: *2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies* (2014), pp. 294–299.

[66] M Gerla, Jui-Ting Weng, and G Pau. "Pics-on-wheels: Photo surveillance in the vehicular cloud". In: *2013 International Conference on Computing, Networking and Communications (ICNC)* (2013), pp. 1123–1127.

[67] Marilena Kampa and Elias Castanas. "Human health effects of air pollution". In: *Environmental pollution* 151.2 (2008), pp. 362–367.

[68] Jean Bousquet, Josep M Anto, Isabella Annesi-Maesano, Toni Dedeu, Eve Dupas, Jean-Louis Pépin, Landry Stephane Zeng Eyindanga, Sylvie Arnavielhe, Julia Ayache, Xavier Basagana, Samuel Benveniste, Nuria Calves Venturos, Hing Kin Chan, Mehdi Cheraitia, Yves Dauvilliers, Judith Garcia-Aymerich, Ingrid Jullian-Desayes, Chitra Dinesh, Daniel Laune, Jade Lu Dac, Ismael Nujurally, Giovanni Pau, Robert Picard, Xavier Rodo, Renaud Tamisier, Michael Bewick, Nils E Billo, Wienczyslawa Czarlewski, Joao Fonseca, Ludger Klimek, Oliver Pfaar, and Jean-Marc Bourez. "POLLAR: Impact of air POLLution on Asthma and Rhinitis; a European Institute of Innovation and Technology Health (EIT Health) project". In: *Clinical and Translational Allergy* 8.1 (2018), p. 36. ISSN: 2045-7022. DOI: 10.1186/s13601-018-0221-z. URL: https://doi.org/10.1186/s13601-018-0221-z.

[69] Dirk Merkel. "Docker: lightweight linux containers for consistent development and deployment". In: *Linux journal* 2014.239 (2014), p. 2.

[70] World Heath Organization. *9 out of 10 people worldwide breathe polluted air, but more countries are taking action.* 2018. URL: http://www.who.int/news-room/detail/02-05-2018-9-out-of-10-people-worldwide-breathe-polluted-air-but-more-countries-are-taking-action.

[71] Catia Prandi, Andrea Melis, Marco Prandini, Giovanni Delnevo, Lorenzo Monti, Silvia Mirri, and Paola Salomoni. "Gamifying cultural experiences across the urban environment". In: *Multimedia Tools and Applications* (2018), pp. 1–24.

[72] A C Pinheiro, C Viegas, S Viegas, C Ver\'\issimo, J Brandão, and M F Macedo. "Indoor Air Quality in Portuguese Archives: A Snapshot on Exposure Levels". In: *Journal of Toxicology and Environmental Health, Part A* 75.22-23 (2012), pp. 1359–1370. DOI: 10.1080/15287394.2012.721168. URL: https://doi.org/10.1080/15287394.2012.721168.

[73] Catia Prandi, Silvia Mirri, Stefano Ferretti, and Paola Salomoni. "On the Need of Trustworthy Sensing and Crowdsourcing for Urban Accessibility in Smart City". In: *ACM Transactions on Internet Technology (TOIT)* 18.1 (2017), p. 4.

[74] UNESCO. *University of Coimbra – Alta and Sofia*. 2013. URL: http://whc.unesco.org/en/list/1387.

[75] Nuno Carlos Pedroso de Moura Correia. "A arquitectura do Paco das Escolas". MA thesis. 2005, p. 65. URL: https://estudogeral.sib.uc.pt/handle/10316/9007.

[76] Luisa Dias Pereira, Adélio Rodrigues Gaspar, and José Joaquim Costa. "Assessment of the indoor environmental conditions of a baroque library in Portugal". In: *Energy Procedia* 133 (2017), pp. 257–267. ISSN: 1876-6102. DOI: https://doi.org/10.1016/j.egypro.2017.09.385. URL: http://www.sciencedirect.com/science/article/pii/S1876610217344764.

[77] UC. *UC — Visiting the UC — Baroque Library*. URL: http://www.uc.pt/en/informacaopara/visit/paco/library.

[78] World Weather Online. *World Weather Online – Coimbra*. 2018. URL: https://www.worldweatheronline.com/coimbra-weather/coimbra/pt.aspx.

[79] James W P Campbell Will Pryce. *The Library: A World History*. Univ of Chicago Pr, 2013.

[80] U S Environmental Protection Agency (USEPA). *Air Now. Air Quality Index Basics*. 2016. URL: http://airnow.gov/index.cfm?action=aqibasics.aqi.

[81] C Arden Pope, Michael J Thun, Mohan M Namboodiri, Douglas W Dockery, John S Evans, Frank E Speizer, Clark W Heath, et al. "Particulate air pollution as a predictor of mortality in a prospective study of US adults". In: *American journal of respiratory and critical care medicine* 151.3 (1995), pp. 669–674.

[82] World Heath Organization. *9 out of 10 people worldwide breathe polluted air, but more countries are taking action.* 2018. URL: http://www.who.int/news-room/detail/02-05-2018-9-out-of-10-people-worldwide-breathe-polluted-air-but-more-countries-are-taking-action.

[83] Abdeen Mustafa Omer. "Energy, environment and sustainable development". In: *Renewable and sustainable energy reviews* 12.9 (2008), pp. 2265–2300.

[84] Daniel A Vallero. "Air pollution monitoring changes to accompany the transition from a control to a systems focus". In: *Sustainability* 8.12 (2016), p. 1216.

[85] PEAN UNION et al. "Directive 2008/50/EC of the European Parliament and of the Council of 21 May 2008 on ambient air quality and cleaner air for Europe". In: *Official Journal of the European Union* (2008).

[86] Silvia Mirri, Catia Prandi, Marco Roccetti, and Paola Salomoni. "Walking under a Different Sky: Urban Colored Routes for Creative Engagement and Pleasure". In: *International Journal of Human–Computer Interaction* 33.12 (2017), pp. 1010–1021.

[87] Silvia Mirri, Marco Roccetti, and Paola Salomoni. "Collaborative design of software applications: the role of users". In: *Human-centric Computing and Information Sciences* 8.1 (2018), p. 6.

[88] Silvia Mirri, Catia Prandi, Marco Roccetti, and Paola Salomoni. "Handmade narrations: handling digital narrations on food and gastronomic culture". In: *Journal on Computing and Cultural Heritage (JOCCH)* 10.4 (2017), p. 20.

[89] Anjali Srivastava and B Padma S Rao. "Urban air pollution modeling". In: *Air Quality-Models and Applications*. InTech, 2011.

[90] Rong Lu, Richard P Turco, and Mark Z Jacobson. "An integrated air pollution modeling system for urban and regional scales: 1. Structure and performance". In: *Journal of Geophysical Research: Atmospheres* 102.D5 (1997), pp. 6063–6079.

[91] Aaron Daly and Paolo Zannetti. "Air pollution modeling–An overview". In: *Ambient air pollution* (2007), pp. 15–28.

[92] Piotr Holnicki and Zbigniew Nahorski. "Emission data uncertainty in urban air quality modeling—case study". In: *Environmental Modeling & Assessment* 20.6 (2015), pp. 583–597.

[93] David J Nowak, Patrick J McHale, Myriam Ibarra, Daniel Crane, Jack C Stevens, and Chris J Luley. "Modeling the effects of urban vegetation on air pollution". In: *Air pollution modeling and its application XII*. Springer, 1998, pp. 399–407.

[94] Saravanan Arunachalam, Alejandro Valencia, Yasuyuki Akita, Marc L Serre, Mohammad Omary, Valerie Garcia, and Vlad Isakov. "A method for estimating urban background concentrations in support of hybrid air pollution modeling for environmental health studies". In: *International journal of environmental research and public health* 11.10 (2014), pp. 10518–10536.

[95] E Wendell Hewson and Lars E Olsson. "Lake effects on air pollution dispersion". In: *Journal of the Air Pollution Control Association* 17.11 (1967), pp. 757–761.

[96] Nelson L Seaman. "Meteorological modeling for air-quality assessments". In: *Atmospheric environment* 34.12-14 (2000), pp. 2231–2259.

[97] C I A CAN. *History of CAN technology*. 2016. URL: https://www.can-cia.org/can-knowledge/can/can-history/.

[98] Craig P Szydlowski. "CAN Specification 2.0: Protocol and Implementations". In: *SAE Technical Paper*. SAE International, 1992. DOI: 10.4271/921603. URL: https://doi.org/10.4271/921603.

[99] ISO. *ISO 11898-1: Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling*. 2015. URL: https://www.iso.org/standard/63648.html.

[100] ISO. *ISO 11898-2: Road vehicles — Controller area network (CAN) — Part 2: High-speed medium access unit*. 2016. URL: https://www.iso.org/standard/67244.html.

[101] ISO. *ISO 11898-3:2006 Road vehicles — Controller area network (CAN) — Part 3: Low-speed, fault-tolerant, medium-dependent interface*. 2006. URL: https://www.iso.org/standard/36055.html.

[102] Olaf Pfeiffer, Andrew Ayre, and Christian Keydel. *Embedded networking with CAN and CANopen*. Copperhill Media, 2008.

[103] CAN-CIA. *Can Frame detail*. 2020. URL: https://www.can-cia.org/can-knowledge/can/can-data-link-layers/.

[104] SAE. *SAE J1850: Class B Data Communications Network Interface*. 2015. URL: https://www.sae.org/standards/content/j1850_201510/.

[105] ISO. *ISO 9141-2:1994 Road vehicles — Diagnostic systems — Part 2: CARB requirements for interchange of digital information*. 1994. URL: https://www.iso.org/standard/16738.html.

[106] ISO. *ISO14230-4:2000 Road vehicles — Diagnostic systems — Keyword Protocol 2000 — Part 4: Requirements for emission-related systems*. 2000. URL: https://www.iso.org/standard/28826.html.

[107] Csselectronics. *OBD2 pinout*. 2020. URL: https://www.csselectronics.com/screen/page/simple-intro-obd2-explained/.

[108] Kazuki Iehira, Hiroyuki Inoue, and Kenji Ishida. "Spoofing attack using bus-off attacks against a specific ECU of the CAN bus". In: *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE. 2018, pp. 1–4.

[109] Hyun Min Song, Ha Rang Kim, and Huy Kang Kim. "Intrusion detection system based on the analysis of time intervals of CAN messages for in-vehicle network". In: *2016 international conference on information networking (ICOIN)*. IEEE. 2016, pp. 63–68.

[110] iWave Technologies. *iWave OBD2*. 2016. URL: https://www.iwavesystems.com/product/obd-ii-device/.

[111] SAE. *SAE J2284: High-Speed CAN (HSC) for Vehicle Applications at 500 KBPS*. 2002. URL: https://www.sae.org/standards/content/j2284/3_200203/.

[112] Nissan. *e-Pedal*. 2018. URL: https://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/e_Pedal.html.

[113] Qingsong Wang, Ping Ping, Xuejuan Zhao, Guanquan Chu, Jinhua Sun, and Chunhua Chen. "Thermal runaway caused fire and explosion of lithium ion battery". In: *Journal of power sources* 208 (2012), pp. 210–224.

[114] Pablo Rodriguez, Rajiv Chakravorty, Julian Chesterfield, Ian Pratt, and Suman Banerjee. "Mar: A commuter router infrastructure for the mobile internet". In: *Proceedings of the 2nd international conference on Mobile systems, applications, and services*. 2004, pp. 217–230.

[115] Joshua Hare, Lance Hartung, and Suman Banerjee. "Beyond deployments and testbeds: experiences with public usage on vehicular {WiFi} hotspots". In: *Proceedings of the 10th international conference on Mobile systems, applications, and services*. 2012, p. 14.

[116] Jakob Eriksson, Hari Balakrishnan, and Samuel Madden. "Cabernet: vehicular content delivery using {WiFi}". In: *Proceedings of the 14th {ACM} international conference on {Mobile} computing and networking - {MobiCom} '08*. San Francisco, California, USA: ACM Press, 2008, p. 199. ISBN: 978-1-60558-096-8. DOI: 10.1145/1409944.1409968. URL: http://portal.acm.org/citation.cfm?doid=1409944.1409968.

[117] Carlos E Andrade, Simon D Byers, Vijay Gopalakrishnan, Emir Halepovic, David J Poole, Lien K Tran, and Christopher T Volinsky. "Connected cars in cellular network: a measurement study". In: *Proceedings of the 2017 {Internet} {Measurement} {Conference} on - {IMC} '17*. London, United Kingdom: ACM Press, 2017, pp. 235–241. ISBN: 978-1-4503-5118-8. DOI: 10.1145/3131365.3131403. URL: http://dl.acm.org/citation.cfm?doid=3131365.3131403.

[118] U S D O T - NHTSA. *AUTOMATED DRIVING SYSTEMS*. 2017. URL: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/13069a-ads2.0_090617_v9a_tag.pdf.

[119] Kathy Winter. *For Self-Driving Cars, There's Big Meaning Behind One Big Number: 4 Terabytes*. 2019. URL: https://newsroom.intel.com/editorials/self-driving-cars-big-meaning-behind-one-number-4-terabytes/%7B%5C#%7Dgs.w8452f.

[120] *Tesla Autopilot*. URL: https://en.wikipedia.org/wiki/Tesla_Autopilot.

[121] Kevin J Ryan. *Tesla Explains How A.I. Is Making Its Self-Driving Cars Smarter*. 2019. URL: https://www.inc.com/kevin-j-ryan/how-tesla-is-using-ai-to-make-self-driving-cars-smarter.html.

[122] Bernard Marr. *The Amazing Ways Tesla Is Using Artificial Intelligence And Big Data*. 2019. URL: https://bernardmarr.com/default.asp?contentID=1251.

[123] Richard Waters. *Why Tesla is taking a different approach to self-driving cars*. 2019. URL: https://www.ft.com/content/b423fe54-6cfa-11e9-80c7-60ee53e6681d.

[124] Liane Yvkoff. *BMW Rolls-Out Its Intelligent Personal Assistant Feature Via Over-The-Air Update*. 2019. URL: https://www.forbes.com/sites/lianeyvkoff/2019/05/30/bmw-rolls-out-its-intelligent-personal-assistant-feature-via-over-the-air-update/#5030b92f5894.

[125] Ka Lok Man, T O Ting, Tomas Krilavicius, Kaiyu Wan, C Chen, J Chang, and S H Poon. "Towards a hybrid approach to SoC estimation for a smart Battery Management System (BMS) and battery supported Cyber-Physical Systems (CPS)". In: *2012 2nd Baltic Congress on Future Internet Communications*. IEEE. 2012, pp. 113–116.

[126] Guan-Hua Tu, Chi-Yu Li, Chunyi Peng, Zengwen Yuan, Yuanjie Li, Xiaohu Zhao, and Songwu Lu. "VoLTE* A Lightweight Voice Solution to 4G LTE Networks". In: *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*. 2016, pp. 3–8.

[127] *OpenSignal - Mobile Network Experience Report October 2019*. URL: https://www.opensignal.com/reports/2019/10/italy/mobile-network-experiencem.

[128] Mansoor Shafi, Andreas F Molisch, Peter J Smith, Thomas Haustein, Peiying Zhu, Prasan De Silva, Fredrik Tufvesson, Anass Benjebbour, and Gerhard Wunder. "5G: A tutorial overview of standards, trials, challenges, deployment, and practice". In: *IEEE journal on selected areas in communications* 35.6 (2017), pp. 1201–1221.

[129] Andreas Festag. “Standards for vehicular communication—from IEEE 802.11 p to 5G”. In: *e & i Elektrotechnik und Informationstechnik* 132.7 (2015), pp. 409–416.

[130] David Xun Yang, Yuchen Guo, and Osama Aboul-Magd. “802.11 ax: The Coming New WLAN System with More Than 4x MAC Throughput Enhancement”. In: *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*. IEEE. 2017, pp. 1–5.

[131] Moussa Ayyash, Hany Elgala, Abdallah Khreishah, Volker Jungnickel, Thomas Little, Sihua Shao, Michael Rahaim, Dominic Schulz, Jonas Hilt, and Ronald Freund. “Coexistence of WiFi and LiFi toward 5G: concepts, opportunities, and challenges”. In: *IEEE Communications Magazine* 54.2 (2016), pp. 64–71.

[132] Yasir Mehmood, Noman Haider, Muhammad Imran, Andreas Timm-Giel, and Mohsen Guizani. “M2M communications in 5G: state-of-the-art architecture, recent advances, and research challenges”. In: *IEEE Communications Magazine* 55.9 (2017), pp. 194–201.

[133] J Ott and D Kutscher. “Drive-thru internet: {IEEE} 802.1 1b for ”automobile” users”. In: {*IEEE*} {*INFOCOM*} *2004*. Vol. 1. Hong Kong, PR China: IEEE, 2004, pp. 362–373. ISBN: 978-0-7803-8355-5. DOI: 10.1109/INFCOM.2004.1354509. URL: http://ieeexplore.ieee.org/document/1354509/.

[134] R. Gass, J. Scott, and C. Diot. “Measurements of In-Motion 802.11 Networking”. In: *Seventh IEEE Workshop on Mobile Computing Systems Applications (WMCSA'06 Supplement)*. 2006, pp. 69–74. DOI: 10.1109/WMCSA.2006.14.

[135] David Hadaller, Srinivasan Keshav, Tim Brecht, and Shubham Agarwal. “Vehicular opportunistic communication under the microscope”. In: *Proceedings of the 5th international conference on {Mobile} systems, applications and services - {MobiSys} '07*. San Juan, Puerto Rico: ACM Press, 2007, p. 206. ISBN: 978-1-59593-614-1. DOI: 10.1145/1247660.1247685. URL: http://portal.acm.org/citation.cfm?doid=1247660.1247685.

[136] Ratul Mahajan, John Zahorjan, and Brian Zill. “Understanding wifi-based connectivity from moving vehicles”. In: *Proceedings of the 7th {ACM} {SIGCOMM} conference on {Internet} measurement - {IMC} '07*. San Diego, California, USA: ACM Press, 2007, p. 321. ISBN: 978-1-59593-908-1. DOI: 10.1145/1298306.1298351. URL: http://portal.acm.org/citation.cfm?doid=1298306.1298351.

[137] Hamed Soroush, Nilanjan Banerjee, Mark Corner, Brian Levine, and Brian Lynn. “A retrospective look at the UMass DOME mobile testbed”. In: *ACM SIGMOBILE Mobile Computing and Communications Review* 15.4 (2012), pp. 2–15.

[138] Vladimir Bychkovsky, Bret Hull, Allen Miu, Hari Balakrishnan, and Samuel Madden. "A measurement study of vehicular internet access using in situ {Wi}-{Fi} networks". In: *Proceedings of the 12th annual international conference on {Mobile} computing and networking - {MobiCom} '06*. Los Angeles, CA, USA: ACM Press, 2006, p. 50. ISBN: 978-1-59593-286-0. DOI: 10.1145/1161089.1161097. URL: http://portal.acm.org/citation.cfm?doid=1161089.1161097.

[139] HyunJong Lee, Jason Flinn, and Basavaraj Tonshal. "RAVEN: Improving Interactive Latency for the Connected Car". In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. MobiCom '18. New Delhi, India: Association for Computing Machinery, 2018, pp. 557–572. ISBN: 9781450359030. DOI: 10.1145/3241539.3241571. URL: https://doi.org/10.1145/3241539.3241571.

[140] Pralhad Deshpande, Xiaoxiao Hou, and Samir R Das. "Performance comparison of 3G and metro-scale {WiFi} for vehicular network access". In: *Proceedings of the 10th annual conference on {Internet} measurement - {IMC} '10*. Melbourne, Australia: ACM Press, 2010, p. 301. ISBN: 978-1-4503-0483-2. DOI: 10.1145/1879141.1879180. URL: http://portal.acm.org/citation.cfm?doid=1879141.1879180.

[141] Hamed Soroush, Peter Gilbert, Nilanjan Banerjee, and Brian Neil Levine. "Concurrent {Wi}-{Fi} for {Mobile} {Users}: {Analysis} and {Measurements}". In: *CoNEXT* (2011), p. 12.

[142] Dorothy Stanley, Jesse R Walker, and Dr. Bernard D Aboba Ph.D. *Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs*. RFC 4017. 2005. DOI: 10.17487/RFC4017. URL: https://rfc-editor.org/rfc/rfc4017.txt.

[143] J. Ott and D. Kutscher. "A disconnection-tolerant transport for drive-thru Internet environments". In: *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.* Vol. 3. 2005, 1849–1862 vol. 3. DOI: 10.1109/INFCOM.2005.1498464.

[144] Xiaoxiao Hou, Pralhad Deshpande, and Samir Ranjan Das. "Moving bits from 3G to metro-scale WiFi for vehicular network access: An integrated transport layer solution". In: *2011 19th IEEE International Conference on Network Protocols* (2011), pp. 353–362.

[145] Zhenyu Song, Longfei Shangguan, and Kyle Jamieson. "Wi-Fi Goes to Town: Rapid Picocell Switching for Wireless Transit Networks". In: *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. SIGCOMM '17. Los Angeles, CA, USA: Association for Computing Machinery, 2017, pp. 322–334. ISBN: 9781450346535. DOI: 10.1145/3098822.3098846. URL: https://doi.org/10.1145/3098822.3098846.

[146] Anastasios Giannoulis, Marco Fiore, and Edward W Knightly. "Supporting vehicular mobility in urban multi-hop wireless networks". In: *Proceeding of the 6th international conference on {Mobile} systems, applications, and services - {MobiSys} '08*. Breckenridge, CO, USA: ACM Press, 2008, p. 54. ISBN: 978-1-60558-139-2. DOI: 10.1145/1378600.1378608. URL: http://portal.acm.org/citation.cfm?doid=1378600.1378608.

[147] Aruna Balasubramanian, Ratul Mahajan, Arun Venkataramani, Brian Neil Levine, and John Zahorjan. "Interactive {WiFi} {Connectivity} {For} {Moving} {Vehicles}". In: *SIGCOMM* (2008), p. 12.

[148] Pralhad Deshpande, Anand Kashyap, Chul Sung, and Samir R Das. "Predictive methods for improved vehicular {WiFi} access". In: *Proceedings of the 7th international conference on {Mobile} systems, applications, and services - {Mobisys} '09*. Wroclaw, Poland: ACM Press, 2009, p. 263. ISBN: 978-1-60558-566-6. DOI: 10.1145/1555816.1555843. URL: http://portal.acm.org/citation.cfm?doid=1555816.1555843.

[149] Alan Ford, Costin Raiciu, Mark J Handley, and Olivier Bonaventure. *TCP Extensions for Multipath Operation with Multiple Addresses*. RFC 6824. 2013. DOI: 10.17487/RFC6824. URL: https://rfc-editor.org/rfc/rfc6824.txt.

[150] Aruna Balasubramanian, Ratul Mahajan, and Arun Venkataramani. "Augmenting mobile 3G using {WiFi}". In: *Proceedings of the 8th international conference on {Mobile} systems, applications, and services - {MobiSys} '10*. San Francisco, California, USA: ACM Press, 2010, p. 209. ISBN: 978-1-60558-985-5. DOI: 10.1145/1814433.1814456. URL: http://portal.acm.org/citation.cfm?doid=1814433.1814456.

[151] Pralhad Deshpande, Anand Kashyap, Chul Sung, and Samir R Das. "Predictive methods for improved vehicular WiFi access". In: *Proceedings of the 7th international conference on Mobile systems, applications, and services*. 2009, pp. 263–276.

[152] Ajay Tirumala. "Iperf: The TCP/UDP bandwidth measurement tool". In: (1999). URL: http://dast.nlanr.net/Projects/Iperf/.

[153] Neal Cardwell, Yuchung Cheng, C Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. "BBR: Congestion-based congestion control". In: (2016).

[154] Sangtae Ha, Injong Rhee, and Lisong Xu. "CUBIC: a new TCP-friendly high-speed TCP variant". In: *ACM SIGOPS operating systems review* 42.5 (2008), pp. 64–74.

[155] *Free France*. URL: https://www.free.fr/freebox.

[156] Tcpdump.org. *tcpdump*. 2019. URL: https://www.tcpdump.org.

[157] Henrik Adolfsson. "Comparison of Auto-Scaling Policies Using Docker Swarm". MA thesis. Linköping University, Database and information techniques, 2019, p. 65.

[158] D Bernstein. "Containers and Cloud: From LXC to Docker to Kubernetes". In: *IEEE Cloud Computing* 1.3 (2014), pp. 81–84. DOI: 10.1109/MCC.2014.51.

[159] Anil Madhavapeddy and David J Scott. "Unikernels: Rise of the Virtual Library Operating System: What If All the Software Layers in a Virtual Appliance Were Compiled within the Same Safe, High-Level Language Framework?" In: *Queue* 11.11 (2013), pp. 30–44. ISSN: 1542-7730. DOI: 10.1145/2557963.2566628. URL: https://doi.org/10.1145/2557963.2566628.

[160] Xiaofei Wang, Yiwen Han, Chenyang Wang, Qiyang Zhao, Xu Chen, and Min Chen. "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning". In: *IEEE Network* 33.5 (2019), pp. 156–165.

[161] S Yang. "IoT Stream Processing and Analytics in the Fog". In: *IEEE Communications Magazine* 55.8 (2017), pp. 21–27. DOI: 10.1109/MCOM.2017.1600840.

[162] M. Amadeo, G. Ruggeri, C. Campolo, and A. Molinaro. "IoT Services Allocation at the Edge via Named Data Networking: From Optimal Bounds to Practical Design". In: *IEEE Transactions on Network and Service Management* 16.2 (2019), pp. 661–674. DOI: 10.1109/TNSM.2019.2900274.

[163] X. Wang, L. T. Yang, X. Xie, J. Jin, and M. J. Deen. "A Cloud-Edge Computing Framework for Cyber-Physical-Social Services". In: *IEEE Communications Magazine* 55.11 (2017), pp. 80–85. DOI: 10.1109/MCOM.2017.1700360.

[164] C Tschudin and M Sifalakis. "Named functions and cached computations". In: *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*. 2014, pp. 851–857. DOI: 10.1109/CCNC.2014.6940518.

[165] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. "Networking Named Content". In: *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*. CoNEXT '09. New York, NY, USA: Association for Computing Machinery, 2009, pp. 1–12. ISBN: 9781605586366. DOI: 10.1145/1658939.1658941. URL: https://doi.org/10.1145/1658939.1658941.

[166] Manolis Sifalakis, Basil Kohler, Christopher Scherb, and Christian Tschudin. "An Information Centric Network for computing the distribution of computations". In: *ICN 2014 - Proceedings of the 1st International Conference on Information-Centric Networking* (2014), pp. 137–146. DOI: 10.1145/2660129.2660150.

[167] Ioannis Król Micha\land Psaras. "NFaaS: Named Function as a Service". In: *Proceedings of the 4th ACM Conference on Information-Centric Networking*. ICN '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 134–144. ISBN: 9781450351225. DOI: 10.1145/3125719.3125727. URL: https://doi.org/10.1145/3125719.3125727.

[168] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen. "In-Edge AI: Intelligentizing Mobile Edge Computing, Caching and Communication by Federated Learning". In: *IEEE Network* 33.5 (2019), pp. 156–165. DOI: 10.1109/MNET.2019.1800286.

[169] Antonio Carzaniga, Michele Papalini, and Alexander L. Wolf. "Content-Based Publish/Subscribe Networking and Information-Centric Networking". In: *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*. ICN '11. Toronto, Ontario, Canada: Association for Computing Machinery, 2011, pp. 56–61. ISBN: 9781450308014. DOI: 10.1145/2018584.2018599. URL: https://doi.org/10.1145/2018584.2018599.

[170] Alexander Afanasyev, P Mahadevan, Ilya Moiseenko, Ersin Uzun, and Lixia Zhang. "Interest flooding attack and countermeasures in Named Data Networking". In: *2013 IFIP Networking Conference, IFIP Networking 2013*. 2013, pp. 1–9.

[171] Ilya Moiseenko, Mark Stapp, and David Oran. "Communication Patterns for Web Interaction in Named Data Networking". In: ACM-ICN '14. Paris, France: Association for Computing Machinery, 2014, pp. 87–96. ISBN: 9781450332064. DOI: 10.1145/2660129.2660152. URL: https://doi.org/10.1145/2660129.2660152.

[172] Valuates Reports. *IoT Sensors Market Size to Reach USD 58.9 Billion by 2026*. 2020. URL: https://www.prnewswire.com/in/news-releases/iot-sensors-market-size-to-reach-usd-58-9-billion-by-2026-valuates-reports-858639909.html.

[173] Precedence Research. *Autonomous Vehicle Market (By Application: Defense and Transportation (Commercial and Industrial))*. 2020. URL: https://www.precedenceresearch.com/autonomous-vehicle-market.

[174] G Grassi, D Pesavento, G Pau, L Zhang, and S Fdida. "Navigo: Interest forwarding by geolocations in vehicular Named Data Networking". In: *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. 2015, pp. 1–10.

[175] Giulio Grassi, Kyle Jamieson, Paramvir Bahl, and Giovanni Pau. "Parkmaster: An In-vehicle, Edge-based Video Analytics Service for Detecting Open Parking Spaces in Urban Environments". In: *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. SEC '17. New York, NY, USA: ACM, 2017, 16:1–16:14.

Ringraziamenti

Mi sembra passato pochissimo da quando pensavo ai ringraziamenti della tesi magistrale, eppure sono passati già tre anni.
Tre anni ricchi di bellissime esperienze che mi hanno accompagnato durante questo lavoro di dottorato: le trasferte all'estero a Parigi, gli esperimenti a Coimbra, i seminari (e le ore di guida!) alla UCLA a Los Angeles e le giornate passate al politecnico di Macao.

Voglio ringraziare innanzitutto la mia famiglia, il cui supporto e amore mi è sempre fondamentale, soprattutto in un anno particolare com'è stato il 2020;
il prof. Pau, che ha deciso di seguirmi anche in questo lavoro, dandomi sempre delle ottime dritte sia dal punto di vista accademico che umano;
il prof. Giusto e il prof. Atzori che hanno collaborato in questi anni da Cagliari;
Sara, che ha reso decisamente migliore questo 2020 con la sua vicinanza, il suo affetto e il suo essere una persona speciale;
Pietro, Mattia e Sofia, che rendono casa un luogo bello e sereno da vivere;
Alice, Andrea e Luca, che si sono pure presi la briga di revisionare questa tesi e di collaborare attivamente ad alcuni esperimenti;
agli amici bolognesi/bresciani/marchigiani che mi hanno accompagnato fino ad oggi e che spero di rivedere il prima possibile;
e a te che stai leggendo questa tesi.

Il mio percorso non si ferma qui, ma senza la vostra amicizia e supporto non sarei sicuramente la persona che sono diventato oggi.

Grazie, come sempre, di tutto.

Davide