Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

Informatica

Ciclo XXVII

Settore Concorsuale di afferenza: 01/B1 - Informatica

Settore Scientifico Disciplinare: INF/01 - Informatica

Distributed Smart City Services for Urban Ecosystems

Presentata da: Luca Calderoni

Coordinatore Dottorato:
Prof. Paolo Ciaccia

Relatore:
Prof. Dario Maio

Tutore:
Prof. Luciano Margara

Esame finale anno: 2015

Abstract

A Smart City is a high-performance urban context, where citizens live independently and are more aware of the surrounding opportunities, thanks to forward-looking development of economy politics, governance, mobility and environment. ICT infrastructures play a key-role in this new research field being also a mean for society to allow new ideas to prosper and new, more efficient approaches to be developed.

The aim of this work is to research and develop novel solutions, here called *smart services*, in order to solve several upcoming problems and known issues in urban areas and more in general in the modern society context. A specific focus is posed on *smart governance* and on *privacy issues* which have been arisen in the cellular age.

Contents

List of Figures

List of Tables

List of Algorithms

List of Protocols

Chapter 1

Introduction

Smart cities are a lively and growing research topic. In recent years, several scientific studies on the subject have been written, presented and discussed. A comprehensive overview of the state of the art in this research field is available in [111]. While some research works focus on a high-level vision of the urban context, and aim at giving us an idea of how a smart future city could appear to the citizen's eyes, others describe a technological model or service for the city.

Over the years many different definitions of *smart city* have been proposed. Nam and Pardo, analyzing a number of these definitions in two closely related works [86]-[87], notice that most works can be classified into different categories as they discuss future smart cities adopting either an architectonic, social or infrastructural point of view, or a combination of the three. However, independently from the chosen point of view, every contribution aims at urban innovation: in the first case trying to design the exterior and tangible part (architecture, ecology, etc.), in the second the relational part (governance, policy and citizen interaction) and in the third by focusing on the infrastructures, mainly technological and internet based, that combine and connect the city intelligent systems. Depending on the approach used, each work adopts the definition that best fits the specific context.

Another theme that is central to the smart city concept and recurs in its definitions is that of citizen interaction. This interaction transforms the citizen from a passive subject into a live actor, integral part of the system [34], [112], which in turn becomes a real, living ecosystem, sometimes called *living lab* [35].

In *Information and Communication Technology* (ICT), the concept of smart city expands that of intelligent and integrated networks such as the *Ubiquitous Sensor Networks* (USN) [54]. In fact, a smart city receives and integrates knowledge from the *Internet of Things* (IoT), the *Internet of Services* (IoS), the *Internet of Energy* (IoE) and the *Internet of People* (IoP) [46], [51]. For instance, the intelligent integration of

data coming from traffic sensors, weather stations and security cameras spread around the city could be used to provide users (such as citizens, ambulances, taxis) with real-time traffic information and route selection based on the current state of the urban route network. Actors of city automation do not generally need to be human. In fact, another interaction common of smart cities is that of city-to-enterprise infrastructure (as opposed to city-to-citizen) [62]. For example, real-time monitoring of the city electrical grid can be used to automatically warn industrial control systems of possible criticalities in the power supply.

In [46], Hernández-Muñoz et al. discuss the issues related to the deployment of massive sensor networks and the definition of reusable services for citizens and administrations. In this research work, the urban environment is also seen as an open innovation platform to perform large scale experimentation, a concept similar to the one of *Living Lab* proposed in [112]. In [98], Park et al. describe a cloud computing model offering various kind of IT services for the citizen. The management and monitoring of the cloud itself is performed using mobile Android OS clients.

The aim of this thesis is to discuss the impact of ICT applied in the smart city context and to present novel related contributions proposed by the author, focusing on new opportunities made available to each city actor (citizens, companies, authorities). In order to understand the challenges introduced by the widespread usage of ICT in modern cities, several research topics and case studies are proposed. In Chapter 2, a focus on ICT services for smart cities is posed. Front-end mobile services and management-oriented services designed for public administration are probably the most perceptible changes in modern cities. To this purpose, an in-depth discussion on location-aware services (both outdoor and indoor) is proposed along with several other types of smart services. The discussion on location-aware services is mainly based on two systems designed, implemented and deployed by the author [23, 21]. In Chapter 3 a focus is then posed on those enabling infrastructures required to run these kind of services. The discussion relies on some research projects based on the design of ICT infrastructures for the urban context, following an engineering perspective. These systems rely on novel solutions proposed by the author and published in scientific journals [24, 37]. Finally, a consistent part of this work is dedicated to the problems introduced by these pervasive technologies with respect to the user's privacy. These topics are discussed in Chapter 4, where the author also proposes some novel best practices and cryptographic primitives aimed at preserve users location-privacy. The lack of privacy in the cellular age should be in fact seriously considered as one of the biggest challenges of the near future.

1.1 Smart City definition

For the purpose of this thesis, we define a smart city as a high-performance urban context, where citizens are interconnected to each others and to the city itself, which provides a constant flow of information, personalized to the user's needs and preferences. Citizens are more independent, more aware of the surrounding opportunities, and benefit from the integrated services that the city offers. The ICT infrastructure is the brain of the city, governs its body and reacts to the circumstances intelligently. A smart city allows new ideas to prosper and new, more efficient approaches to be developed in economy, politics, governance, mobility, environment and all the other aspects of city life. Following this definition, we model the functioning of a smart city in a three-layers scheme, as shown in Figure 1.1. The three-levels ecosystem gathers data from the world (sensor networks, world wide web and user contribution), stores and processes these data into a kernel (the city brain), and exposes through several services the processed data. These services can be designed to be used by citizens, but also by non-human actors (for instance, in the case of direct communication with industry automated subsystems). At the *access layer*, we have the sources of information for a smart city: sensor networks, relevant knowledge bases made available over the Internet, and the citizens themselves through user contribution. This information is processed at the *kernel layer*, where raw data are gathered and then elaborated by the computing infrastructure (*kernel*) of the city, the city brain. Citizens, and other relevant city actors (such as businesses, industries, local government) query the system and access information through dedicated services and interfaces (the *service layer*).

In this fast-growing context it is predictable to see billions of machines interconnected in the near future. Thus, *Machine-to-Machine* (M2M) communication represents a relevant topic in the smart city field. M2M protocols and architectures were proposed and widely discussed by *European Telecommunications Standards Institute*[1].

1.2 Smart cities today

Smart cities are close to becoming a common reality. Recently, many experiments and pioneering projects have been successfully run. Some of these works cover specific applications and services suitable for smart cities (often involving mobile devices), while other are integrated plans for an entire city, both theoretical or actually implemented.

[1] http://www.etsi.org

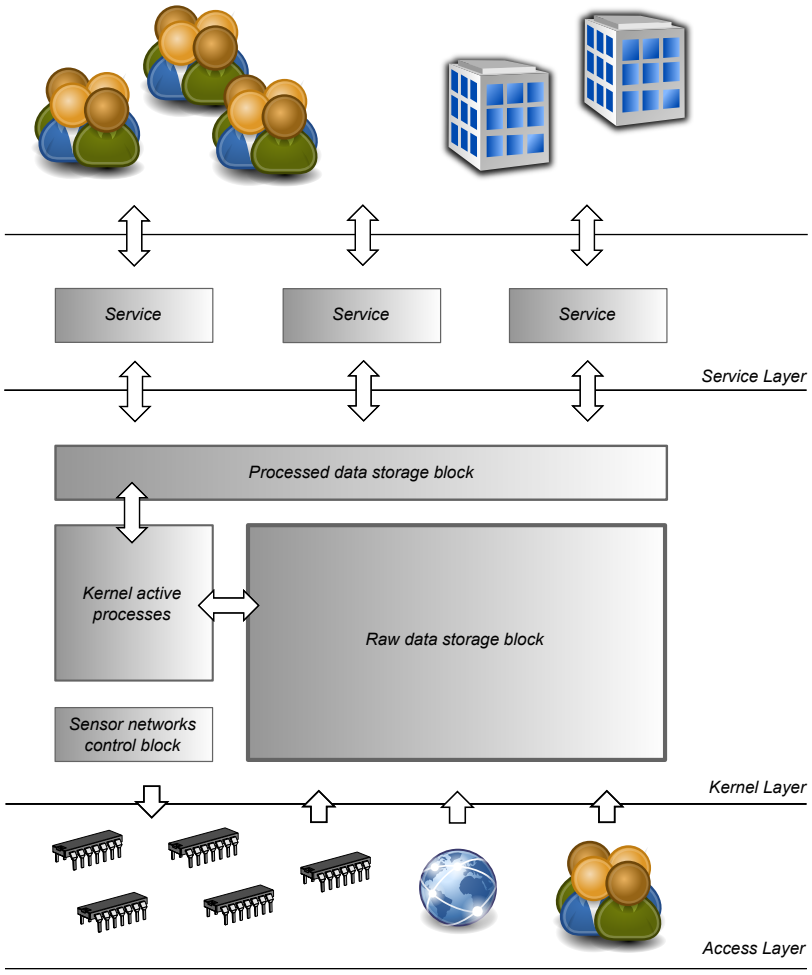


Figure 1.1: A schematic high-level representation of the underlying infrastructure of a future smart city.

An example of the latter is the plan for the construction of an entire city outside Abu Dhabi: *Masdar City* [44]. The city will cover an area of 6 square kilometers and be home to about 50.000 citizens. A major feature of the project is an integrated public transport system, that will minimize the need of private vehicles.

In Zaragoza, Spain, a wide sensor network controls approximately 90% of urban routes. A centralized control point monitors in real-time the traffic of the whole city, and helps the government to improve the road network with adequate policies [44].

Another Spanish city, Santander, is home to one of the most ambitious experiments, that transformed the city into a real living laboratory. *Smart Santander* [44]-[46] represents a true Internet of Things case study [51], housing thousands of devices that form a number of sensor networks. The project collects data from the urban environment and makes the information gathered available to research groups

around the world. The aim of this project is to help design experimental services like traffic and parking remote control, tourism guidance through mobile devices or video monitoring of sensible areas.

The European Union is particularly active in supporting the development of medium-sized cities, contrary to current research that tends to focus on the *global* metropolises. Through the ranking of 70 European cities, the EU aims at sparking competition and therefore help optimize the internal organization and resource allocation practices of those cities and, ultimately, improve living standards [39]. A recent study provides background information on Smart Cities within EU and explains how existing mechanisms perform [26]. In this work the authors propose a custom definition for a Smart City and the map those cities fitting this definition across the Member States. Through an in-depth analysis of the whole context, this report points out that Smart City objectives should be more explicit, well defined and clearly aligned to city development, innovation plans and Europe 2020 in order to be successful.

There is a large number of Living Labs in Europe with a variety of different characteristics. Coordinating and aligning the activities in the domains of Future Internet research and of user driven open innovation was the main goal of a recent funded project named FIREBALL[2].

Intelligent city strategy and planning are at the basis of the definition and evaluation of benchmarks designed for city rankings. In a recent book [64], Nicos Komninos studies the drivers and architectures of the spatial intelligence of cities, the planning processes for intelligent cities and how cities should manage the drivers of spatial intelligence, create smart environments, mobilize communities, and offer new solutions to address city problems.

The *Cassa Depositi e Prestiti* (CDP) published an interesting report where 79 cities (from all around the world) are compared between each other with several *smartness* indicators [25]. It is important to note that one of the most interesting aspects of the smart city field is how to evaluate cities' *smartness*. Today, a lot of governments and institutions need to assign funds, to award some prizes, to raise a city to a leading model, basing each decision on some kind of assessment operated on the concerned cities. It is therefore crucial to dispose of some form of indicators able to express concisely the city features. Defining and computing these indicators could be very complex and should require the participation of experts. In fact, a single indicator could vary depending on social, economical and technical aspects and

[2]http://www.fireball4smartcities.eu/

each of them could depend on several other factors in turn. Hence, this landscape represents an interesting and meaningful topic for future research.

1.3 Smart services for the urban context

The *Smart City* model is strictly related to the ICT infrastructures that the city itself shares with the citizens; these innovative architectures provide a more efficient conservation of the fixed heritage and a better standard of living, thanks to modern and full-accessible services.

These are some research field examples:

- Video control and intelligent urban surveillance
- Security and geocoding in emergency call service
- Energy saving
- Remote automated control of industrial facilities
- Remote hydrological networks and gas supply monitoring
- Logistics
- Safety at work
- Personal and community healthcare
- Waste management and trackback
- Public transport networks
- Automated recognition of road signage and traffic sensors
- Intelligent parking infrastructures
- Monitoring air and noise pollution
- Identification of climate risk areas and zones at risk of natural disaster
- Wired and wireless infrastructure for the web
- NFC and other integrated payment methods
- Digital signage and information retrieval for citizens

- Social interaction
- Tourist facilities
- Digital identity documents management for smart governance
- Electronic payments management for smart economy and tax evasion detection
- Secure digital voting systems for smart governance

In the following, some of the hereabove mentioned services are discussed in more depth. To this purpose, two novel systems designed by the author and available in literature are presented in Chapter 2.

1.3.1 Traffic monitoring

A relevant aspect when evaluating the city *smartness* is related to the innovative approach to urban traffic management.

Today we are witnessing an increasing traffic congestion in the urban context. This is mainly due to both the tendency for people to move to the city from neighboring areas and an increase in the average age of the population, as a result of medical, social and cultural advances. It is estimated that more than 50% of the population is concentrated in cities [85].

This phenomenon of congestion along with many other social, political and economic factors makes urgent a rationalization of urban processes to improve the quality of life, energy saving and more generally the sustainability of the planet [111].

End-user services designed to monitor and plan urban routes in a city context are a relevant and insightful type of smart service. These services usually rely on a back-end which processes raw data gathered by a sensor network.

Thus, ICT infrastructures play a key-role as they are often at the basis of the levers that allow the application of management control to the urban context, in its broadest sense. In particular, sensor networks [51] are the tool that allow to establish a direct and capillary connection between the control and monitoring centres of the city and the entire urban area. An in-depth discussion about ICT enabling infrastructures for end-user services relating urban traffic is provided in Chapter 3.

1.3.2 Weather sensing

Another significant type of service in a modern city relates to weather. Cities need to constantly monitor weather to anticipate heavy storm events and reduce the impact of floods. Information describing precipitation and ground conditions at high spatio-temporal resolution is essential for taking timely action and preventing damages. Traditionally, rain gauges and weather radars are used to monitor rain events, but these sources provide low spatial resolutions and are subject to inaccuracy. Therefore, information needs to be complemented with data from other sources: from citizens' phone calls to the authorities, to relevant on-line media posts, which have the potential of providing timely and valuable information on weather conditions in the city.

However, this information is often scattered through different, static, and not-publicly-available databases. This makes it impossible to use it in an aggregate, standard way, and therefore hampers efficiency of emergency response. In order to reach such an integration an appropriate ICT infrastructure is needed. Weather monitoring infrastructures are described and discussed in Chapter 3.

1.3.3 Electronic payments management for smart economy and tax evasion detection

The level of evasion depends on a number of factors, one of them being fiscal equation. People's tendency to evade income tax declines when the return for due payment of taxes is not obvious. Evasion also depends on the efficiency of the tax administration. Corruption by the tax officials often render control of evasion difficult. Tax administrations resort to various means for plugging in scope of evasion and increasing the level of enforcement. These include, among others, privatization of tax enforcement, tax farming and institution of Pre-Shipment Inspection (PSI) agencies. In 2011, HMRC stated that it would continue to crack down on tax evasion, with a goal of collecting Â£18 billion in revenue before 2015. Automated systems capable of collect large amount of economic transactions and cross them with personal citizens' information in order to detect possible tax evasion issues are therefore of big interests for modern governments.

1.3.4 Digital identity documents management for smart governance

The recent introduction and spread of electronic identity documents offer a good example of what a smart service could be. When each citizen is provided with a

personal and unique identification number on a digital device, a wide number of government-to-citizen services can be set up. The introduction of services such as automated certificates printing or automated border control are just some of the conceivable or already available services.

The Italian Electronic Identity Card (EIC), is a personal identification document that is replacing the paper-based ID card in Italy.

The Ministry of Internal Affairs supplies the required network infrastructure, software updates and security architecture. As a consequence the required issuing system is more complex than centralized ones used by other countries or for other electronic cards. Recently many municipalities joined the EIC system and currently about 180 municipalities are equipped to issue EICs. The system allows smaller municipalities to collaborate with nearby larger ones in order to reduce printing cost. The potential user base is estimated at about 50 million.

The Italian EIC is intended for both online and offline identification. Therefore, apart from the printed information, data for identification are stored on a microchip as well as a laser band. Specifically, the microchip contains a digital certificate for online authentication and (optionally) a certificate for digital signatures. The Italian EIC is explicitly designed to give access to e-government services and will become the standard for access to online services offered to Italian citizens by public authorities.

Since the introduction of electronic documents, several attacks were presented and demonstrated and so, a really interesting research field is the one that involves the techniques and the protocols used to avoid these attacks and to make more efficient and useful these documents. This research field is strictly connected with governments and is of real relevance for the smart governance component of future cities. This concepts are expanded and discussed in Chapter 4.

Chapter 2

Location-aware Services

With *location-awareness* we use to refer to the capability of a device or application to deliver information about its physical location to another user or application. The term is most often used in reference to mobile communication devices. A device's location is usually determined by one of three methods: by GPS satellite tracking, by cellular tower triangulation, or by the device's media access control (MAC) address on a Wi-Fi network.

The possibility of collecting location- and time-aware data from the urban context is a particularly interesting feature of a smart city. Those data are in fact fundamental in order to offer high value services to the citizen [20], and they can even serve the purpose of analyzing and understanding the inner dynamics and functioning of the city - social, economical and even psychological [104]. For instance, location- and time-aware data have been used for urban security, usually involving video control and urban safety applications, and for traffic monitoring and optimal route finding, often associated with automated recognition of road signage and traffic sensors [15], [137].

2.0.5 Outdoor positioning

Outdoor positioning is the most common form of location-aware service. To be aware of the position of a mobile device in the urban context, allow to perform several related actions. For instance, some location-aware advertising could be delivered to the user's phone. Tourist information, traffic management and emergency management are just some of the other related aspects. The increasingly frequent use of this kind of services also lead to several privacy concerns. These issues are discussed in Chapter 4.

In a recently published paper [23] the author discussed a case study concerning an *around me* application, that provides users on the go with information about their

current location and its surroundings. The information collected by the system and served to the users is a catalog of interesting objects located within the city. By "interesting objects" we mean shops and businesses, touristic landmarks or useful public services such as hospitals and pharmacies. However, depending on the application, also traffic sensors, weather stations, wireless hotspots, or any other type of point of interest (PoI) can be included. Each object is classified into a specific category, and its geographical location and other relevant information are stored. The user performs searches through a location-aware mobile application, that displays search results as a list of objects, chosen according to the user's position and preferences. The proposed system integrates information coming from pre-existing knowledge bases and user generated content. In fact, a user can suggest interesting objects and shape the system around her own needs.

The system's kernel is a distributed computing infrastructure based on a clustered database management system (DBMS). The external interface is provided by a web service, that users can query though a specific software developed for mobile phones. This platform is selected in order to offer a high degree of flexibility to users on the move, that are the typical target of this kind of application. Moreover, modern mobile phones (correctly called smartphones), are equipped with all the tools needed to realize a location-aware application, such as a GPS receiver and Internet connection.

A first experimental implementation of the service is currently being deployed in Cesena (Italy). Results from this experiment will be used to improve the framework of the service for future releases.

The main point of interest concerning this kind of application is the whole design and implementation problem lying behind. In the following the adopted approach is discussed in details.

2.0.6 Design and architecture of a location-aware application

The mobile application presented as a case study helps users (citizens) to locate useful objects around their current location. The information is retrieved by querying the server infrastructure, and is presented ordered by distance from the user. Distances are updated during the user's movements. In the design of the application and of the computing infrastructure behind it, a pragmatic, practical approach is adopted, based on the integration of existing technologies and newly proposed solutions.

This case study application let us discuss the challenges involved in creating a location-aware mobile service based on live information coming from the city IT infrastructure. The service has been designed with the goal of being a general model

for future applications. In particular, the following discussion focuses on location-aware and mobile development, cloud and cluster based geographical data storage, and spatial data computation. As for each of these topics the author provides implementation and deployment solutions based on currently available technology, this proposal represents the first comprehensive, scientific study on the subject.

Let us imagine a person visiting a city for the first time. She has no idea of what the city has to offer, but she is interested in finding out about museums. With a mobile device on her (phone or tablet) and an Internet connection, she can use this application to retrieve in a few seconds all the surrounding museums. For each of them, she can than request additional details like the entrance cost, opening hours and so on. But the application is not limited to touristic information. Instead, it can display a large variety of objects, to best satisfy the user's needs in every context. For instance, a collection of pharmacies, hospitals and medical centres is useful even for citizens in their own city.

In the design of this solution, however, there is no restriction to a specific city or geographical region. In fact, the provider side that serves the client application is designed in order to be fully scalable, and to be able to serve requests that range from a single city to all over the world.

In our test case application, a number of well-known categories of common urban objects are considered, such as bookstores, cinemas, gas stations, supermarkets and so on. For each category, the provider of the service retrieves relevant information for all of the objects around the world, geo-locates them and fills the database with this information. If we consider an instance of the application covering a small city, objects could be manually entered by the staff. If instead we consider a wider context, an automated solution is needed. Object information and addresses can be crawled from the world wide web, for example considering only well-known address directories in order to find consistent results.

User involvement is a fundamental component of applications designed for a smart urban context [20], [35], [111] - [112]. In the proposed application users are encouraged to submit new objects and categories, or improvements and corrections to the existing ones. For instance, a user could tag a well-formed object in a web page, telling the crawler to consider it when searching for addresses. A community of citizens that insert or tag objects in many ways is considered as well, for example from a web browser interface or directly from the mobile application. A reward system could prompt users to participate.

The proposed solution is based on a client-server architecture, where the clients are mobile devices that interact with a distributed server infrastructure. The structure of a dedicated mobile application and the corresponding remote interface is outlined in Section 2.0.6.2, where design specifications and the complete components diagram are provided, and the client-server architecture is discussed. The provider side is designed with performance, scalability and geographical distribution in mind. Since the system should serve requests coming from all over the world, the provider-side infrastructure has to be able to serve billions of requests per day and perform queries on massive amounts of data. Deployment and implementation are discussed in depth in the following, in Section 2.0.7.

2.0.6.1 Stakeholder analysis

Given the social relevance of any project that is developed for a city and that will shape its future life, an analysis of the stakeholders is particularly important. Table 2.1 shows the different actors involved in the project, the phases of the project development (project outline, design, implementation, operation, maintenance) in which each of them will take part and their interests in pursuing the project.

Like most smart city innovations, the proposed system is designed to improve the citizens fruition of the city. Therefore, the local government will benefit from an increased citizen satisfaction, and a better attractiveness to tourism. Tourists themselves, as well as the inhabitants of the city, will be the final users of the service. The users will be also prompted to directly participate in the project, by suggesting objects or categories to be included in the application. A reward scheme (even monetary) can be put in place to stimulate citizens engagement. The service provider, that is, the entity that will actually develop the system and take care of its maintenance once in operation, will act in close coordination with the local government, especially in the design phase, in order to identify the specific needs of the city. The project could also be run internally by a dedicated team part of the local government. The local government will also help attract external funding originating from private and public investors and innovation initiatives (see Figure 2.1). A business that is listed in the application database will profit indirectly from the application, thanks to the increased visibility to potential customers. The economic viability of the project is assured by the diversified source of income for the service provider: government participation in the initial funding and operation expenses and sponsoring of the application from the listed business. Advertisement revenues can be collected

Actor	Involvement	Benefit
Local Government	Opportunity/needs assessment, concept proposal	1. Increased citizen satisfaction 2. Attracting innovation investments
Citizens	User contribution during operation phase	1. Better urban experience 2. Earning opportunities due to user participation
Businesses	Sponsoring during implementation and operation phase	1. Increased business due to wide information diffusion 2. Better customer experience thanks to location-aware advertising
Tourists	Service end users	1. Better urban experience
Service provider	System design and development, service maintenance	1. Strong business enforced by diverse sources of income (government, businesses, other advertisers)

Table 2.1: For each stakeholder, an analysis concerning which phases of the system lifecycle involve him is proposed. The main benefits deriving from the projects, with respect to each stakeholder, are considered as well.

from businesses that wants to improve their placement in the results proposed by the application to the user.

2.0.6.2 Client-server architecture

In this section a detailed scheme representing the architecture of the entire application, designed according to UML (Unified Modeling Language) standards, is proposed. UML is a visual language for system design and engineering and for object oriented software modeling. In the UML component diagram shown in Figure 2.2 we can see that the client side requires an interface, which is exposed by the provider side. This interface is developed as a web service that serves each client request after authentication.

The client consists in an application for mobile phones. The user interface of the application is composed of two main views. The first handles the objects list, while the second shows the details of a single object. The object list displays, for each object, a short name and the distance from the user. Object categories are identifiable thanks to a representative icon. A user can define custom settings for the

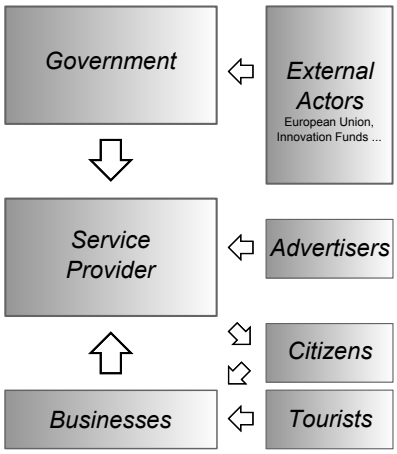


Figure 2.1: An hypothesis of the cash-flow diagram among the stakeholders.

application through a settings view. In particular, he can specify the username and password needed to access the service, as well as select from a list the desired object categories to retrieve and the action range that should be used when the application looks for objects.

The server side handles clients requests via an HTTP interface. After the client is recognized (via the authentication module) the server process the request on the database server. Then, the response is delivered to the mobile client, again through HTTP.

Responsiveness is an important factor in real-time services, so it is advisable to serve each request from an application server near the client. Another requirement of the system is scalability of the server side: thousands of real time requests among millions of objects need a very powerful OLTP architecture in order to be quickly served. Finally, a database with as many records as described and a complex data distribution should present some solution for enforcing data availability and to ensure data recovery in case of a server failure. Therefore, the server side is designed as a geographically distributed cluster architecture with data replication.

2.0.7 Infrastructure for a distributed location-aware application

While the previous sections deal with the high-level design of the system, in the following the infrastructural component is discussed. A discussion concerning the chosen mobile platform, the DBMS used and the cloud framework is proposed. A flexible and scalable solution for a geographically distributed service is proposed as well. Scaling a database as a service (DBaaS) to serve large amount of requests is a

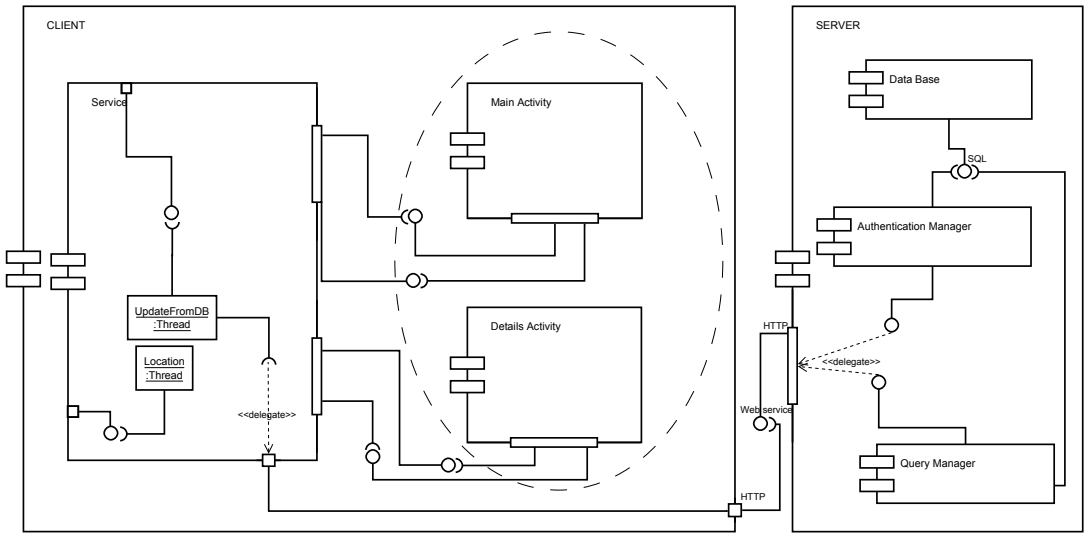


Figure 2.2: The components diagram of the client-server architecture, in the UML standard. The client side is designed as a mobile application for the Android operating system; the server side provides contents to mobile clients through a web service.

crucial objective, as explained in [29]. In order to obtain a high-performance OLTP infrastructure the provider side is deployed on a cluster. Note that the results here presented are related to a single server solution deployed during this research. Details concerning a distributed architecture are provided for completeness and in order to draw some guidelines for future research and implementation.

The server side deployment and implementation are provided in sections 2.0.7.2 and 2.0.7.3. In this complex scenario many practical, scientific and technical problems occur, as for instance efficiently selecting spatial objects around the user from an enormous amount of data. In previous works published on the subject, such as [78] and [82], the area surrounding a geo-located point is first treated as a square having the half-side equal to the search radius. This square is identified by a minimum and maximum longitude and a minimum and maximum latitude as well and is often called the *bounding box*. In this thesis a novel technique for range query optimization is proposed, with improvements in the bounding box calculation (see Section 2.0.7.5). In Section 2.0.7.6, results from an experiment run on database table indices for spatial data are presented. Finally, in Section 2.0.8, a discussion about security and privacy issues for a distributed database and the user application is proposed along with some possible solutions.

2.0.7.1 Android client

We choose to build the client application on the Android platform. In fact, the Google mobile OS holds a large smartphones market share [122]. A typical Android application is composed of activities and services. An activity is the basic application component that provides a view for the user interface, while a service is an application component that does not provide any user interface, but performs background tasks.

This Android application stands on three key components, as shown in Figure 2.2: two activities (main and details) and a service. A welcome activity and a settings activity were also developed in order to provide a graceful application launch and a reliable authentication and preferences handling.

The main activity shows the object list and let the user perform an update from the server. It is the most important application view and it also contains a menu entry for the settings activity.

The details activity shows a single object view, when the user wants to get a more accurate information on the object itself.

All the data are provided by an Android service, which is the most complex component; this service runs separate threads to handle location listening (for GPS or network coordinates retrieval) and to query the provider side. Every single remote request is performed by a dedicated thread. That way, the user interface responsiveness is increased, since each task runs asynchronously. The location thread runs for all the application life cycle and controls two listeners, one for GPS signals and one for network-based location information. Each listener is set to search for new coordinates every 30 seconds; when new coordinates are available, the thread chooses to send or not to send them to the service handler. If the service is not geo-located the new coordinates will always be sent, if the service is already geo-located, the more accurate GPS coordinates will be preferred. If the location thread does not receive new coordinates updates within a certain time interval, it sends the service handler a notification of lost signal. When the signal is lost the user can not update the object list but can ask for displayed object details.

The remote connection thread controls data flow from and to the provider side. It prepares the requests retrieving information from the application settings (such as username, password, bounding box and so on) and send them via an HTTP client. The parameter list is provided in Table 2.2.

The server side will provide a JavaScript Object Notation (JSON) formatted response containing the results. Some error responses such as "username not valid" or "password not matching user" are provided too. In order to handle lighter data on

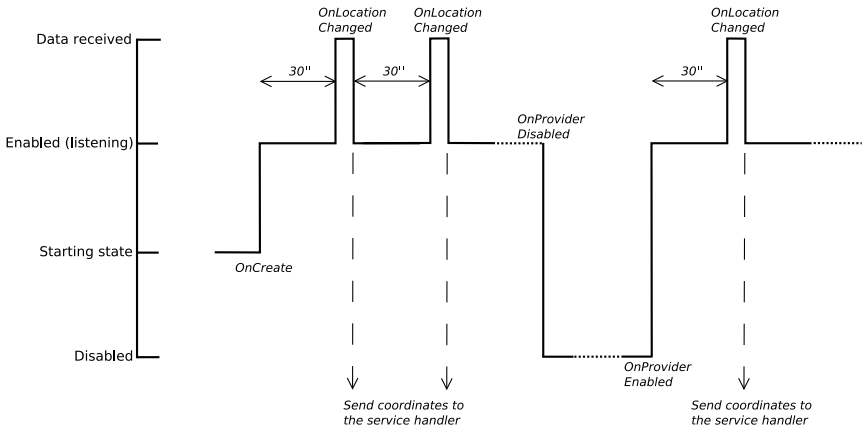


Figure 2.3: The timing logic behind the location thread; when gps or network listeners are enabled on the Android client the thread sends new coordinates to the Android application service every 30 seconds.

the mobile device, when asking for an object list a few data fields are considered (an object identifier, a name, a category and the object coordinates); these values are used to build a `LightObject` array that will inflate the Android ListView. The entire object data will be returned from the server only when the client asks for a single object details.

The Android service is connected to the application activities through Android Messengers (designed as double-sided interfaces). The service internal threads communicate via a single-side Android Handler interface instead.

2.0.7.2 Mirroring & cloud

A key requirement of the designed infrastructure is scalability. The potential user base goes from the inhabitants of a single city, to those of the whole world. While in the former case a local server infrastructure might be sufficient to handle the requests from all the clients, in the latter a distributed solution is needed. Available technology, such as cloud computing or mirroring and clustering capabilities of MySQL, makes this possible [91]. In particular, three goals need to be achieved. First, all clients should be able to query a server that is not physically distant, in order to decrease response time and enhance the user experience. Second, data replication (mirroring) is advisable, in order to ensure a higher fault tolerance. Third, the system should rely on a single, integrated infrastructure, with centralized management, instead of a number of local, disconnected entities.

Field	Description	Values details
`username`	the username	String
`password`	the SHA-512 hash of the password	String
`job_id`	the remote request type	1: object list, 2: single object, 3: category list
`object_id`	the single object identifier	Int (0: no single object requested)
`filter`	the object categories we want to consider	Int[] (0: every category is considered)
`min_lon`	the bounding box minimum longitude	Float(10,6)
`max_lon`	the bounding box maximum longitude	Float(10,6)
`min_lat`	the bounding box minimum latitude	Float(10,6)
`max_lat`	the bounding box maximum latitude	Float(10,6)

Table 2.2: The parameters sent from Android clients to the provided http service in order to perform a remote request.

In this test case, one of the available cloud computing services called Amazon Elastic Compute Cloud (Amazon EC2)[1] is selected. EC2 is a web service that provides resizable computing capacity in the cloud. In practice, it offers an integrated computing service from data centres all over the world, which are collectively referred to as the *cloud*. EC2 servers are geographically distributed in many different regions (East and West US, EU, Asia Pacific and South America), which allows us to achieve goal number one, without compromising goal number three.

In order to have a mirrored and centrally controlled infrastructure (goals two and three), MySQL Cluster is adopted. MySQL Cluster is an open source transactional database. It is designed to be a distributed, multi-master architecture without single points of failure. In particular, the proposed application can take advantage of the possibility to scale horizontally on different servers. Another key feature of MySQL cluster is its ability to perform automatic data partitioning (sharding) with load balancing and the possibility to add nodes to a running cluster, that allows a great amount of scalability.

To explain how the proposed system can benefit from the use of Amazon EC2 and MySQL Cluster, an example of a simple distributed infrastructure, shown in Figure

[1] http://aws.amazon.com/ec2/

2.4 is discussed.

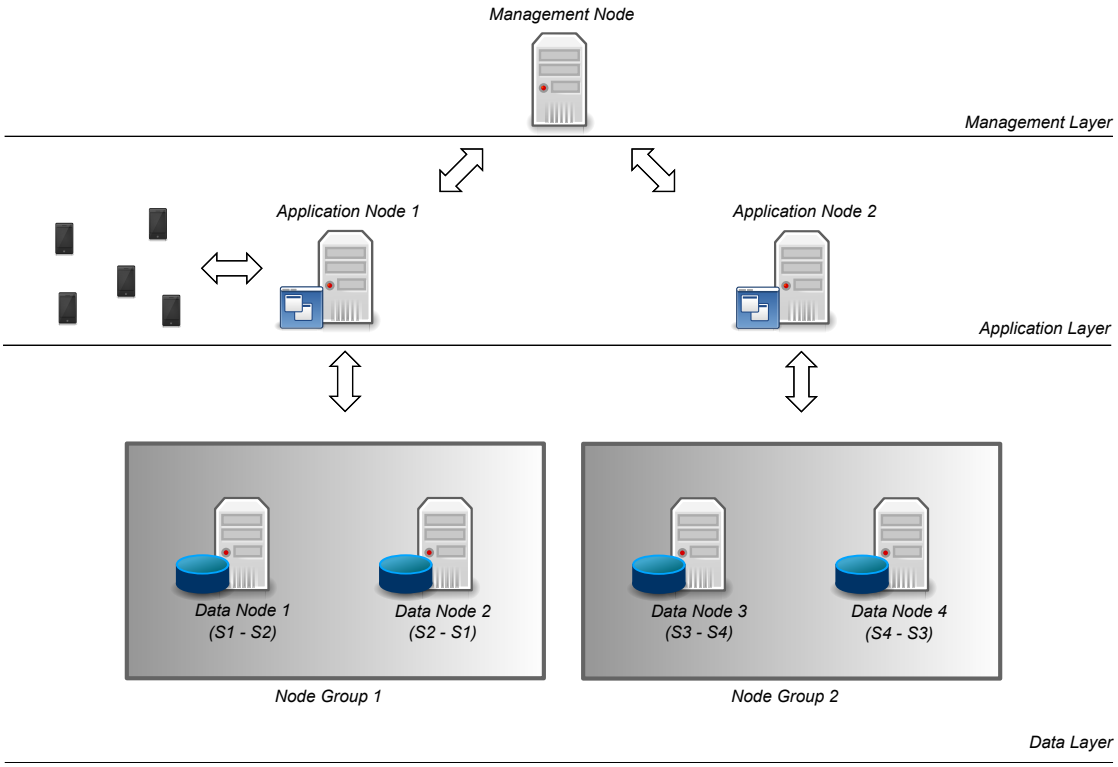


Figure 2.4: The provider side infrastructure in a distributed context. Data base logic is horizontally partitioned in shares (here S1, S2, S3, S4). Each zone is served by one application node and two data nodes in order to reach data distribution; mirroring is provided by slices replication within each node group.

We have a three-levels scheme, where different nodes (servers) play different roles. Data nodes are where the information is stored; application nodes are where the web interface resides, get queried by the clients and act as an interface between them and the data nodes; and management nodes, used for the management and monitoring of the cluster.

In the management layer, the cluster configuration is stored on a set of management nodes. The configuration specifies node roles (management, application or data), their unique id and the tcp/ip connection rules between the nodes. The maximum number of nodes for the reference version of MySQL Cluster is 255.

Application nodes expose the web service that serves client requests, and is composed of a set of geographically distributed servers. The client application has a built in list of a subset of these servers, called *entry nodes* (updates of the list are managed through the automatic updates of the application). Each entry server is assigned to a

wide geographic area, but all of them can handle requests regardless of the client position. When the client needs to retrieve information, it tries to connect to the entry server associated with its current position. During this first request, the entry server can suggest a different server to use for subsequent queries, favoring servers that are geographically closer or subject to a lighter workload. In case of failure of any node, the client will temporarily exclude it from the list and interrogate a different entry server. All this happens transparently to the user.

On the data nodes, the most important information is stored in the `objects` table. To ensure the safety of the data, the information is replicated on different nodes. Nodes are also divided into groups, selected according to their geographical location. MySQL Cluster provides an *auto-sharding* function that let us divide data between data nodes. Usually a primary key hash criteria is used to balance the workload among data nodes, while in this case a *sharding* (division) based on the longitude and latitude fields is preferable, in order to obtain a range partitioning where each data node contains a cluster of geographically-related records.

Let us say, for example, that the application serves citizens of two different cities, one in Canada and one in South Korea. In this case, data nodes will be dislocated in both regions, and divided into two different groups, based on their proximity. Each group will store only local information, that is, information regarding the area they serve. At the group level, the `objects` table is further partitioned a number of times equal to the number of nodes within a group. Let us call this shares of information $S1$, $S2$ for Canada and $S3$, $S4$ for Korea (suppose, for simplicity, that each group is composed of two nodes only). Each data node stores all the information of the data group (to ensure data replication), but for each partition S only one node will act as *master*, meaning that it will be responsible of data management. The other nodes will act as *slaves*. So, in this example, the first node of the Canadian group will store $S1$ as master and $S2$ as slave, while the second will store $S1$ as slave and $S2$ as master. In case of failure, the group is able to continue operating as long as at least one node is still running.

One application node in Canada will receive the queries from local mobile users, and will request the needed information to the local data group. The same happens in Korea. In the case of an application node becoming unavailable, the requests are dynamically and transparently redirected to the other node. This will have an impact on the responsiveness, but the service will still be fully functioning. All the management and monitoring is done from the dedicated management node. The geographical position of this node is irrelevant to the functionality of the system.

2.0.7.3 Database

Since queries need to be as simple as possible, and unnecessary `JOIN` operations should be avoided, the database structure looks quite simple. It includes some tables serving the authentication procedure (stored with the InnoDB engine in order to support transactions), the table `object` that contains objects data and the table `category` that contains object's categories. These two tables are stored with the MyISAM storage engine to improve query responsiveness.

In the distributed context described in Section 2.0.7.2 tables are stored with the NDBCluster storage engine; query responsiveness still stands (principally due to the fact that NDBCluster loads every data entry in the server memory) as transaction support, even if limited to *read committed* isolation level [132].

The large amount of fields of the table `object` is due to the fact that it stores objects of all the different types. This table is queried for a small amount of fields when asking for an objects list (the complex scenario), and for all the table fields when asking for a single object by id (the simple scenario). That way, a list job will not produce a massive response to the client.

Let us focus on tables join (upon the ref key `categotyId` that links objects with their categories) in the complex object list scenario: we want to avoid any kind of join between the two tables to speed up our queries. This is accomplished thanks to the class-factory included in the Android client: this class knows the logic of the categories database identifiers and can map every retrieved object to the correct category directly on the client mobile device, using the `categoryId` value included in table `object`.

One of the distinguishing traits of a smart city is the exposure of on-demand intensive services to the urban actors. This usually implies a dedicated On-Line Transaction Processing (OLTP) distributed infrastructure that can ensure the highest possible reliability. This is usually implemented using the concept of DataBase as a Service (DBaaS) [29]. In this work, the problem is addressed with a new, integrated approach based on cloud computing and database clustering. Since mobile services in the urban context often need some form of geo-location of the clients, in order to provide location-aware services, a common research topic is efficient processing of spatial information. In [82], Mondal et al. propose a new database index structure, designed to improve the efficiency of spatial queries. An original solution to the problem is proposed in Section 2.0.7.6.

2.0.7.4 Interface to the DB

In order to avoid the need of a direct connection to the DB, which may not be possible on mobile Internet connections, the server side exposes an HTTP interface. The data are transferred using JSON objects, that are generated on the fly through a PHP script on the server. The communication happens through HTTP POST requests, one of the standard request mechanisms specified by the HyperText Transfer Protocol (HTTP). An object list request performed by the client will result in the server outputting a JSON formatted string containing the values shown in Table 2.3. These values are then used to fill the object list that appears in the user interface of the Android client.

Field	Description	Values details
`id`	the object identifier	BigInt(20), unsigned
`name`	an object description	Varchar(60)
`categoryId`	the object's category identifier	Int(10), unsigned
`longitude`	the object's longitude	Float(10,6)
`latitude`	the object's latitude	Float(10,6)

Table 2.3: The parameters sent from the server to the Android client after an object list request. Data are Json formatted.

2.0.7.5 Range query optimization

In order to obtain a powerful OLTP architecture, the execution time for the object list retrieval query needs to be minimized. Instead of including real distance evaluation within the query itself, as proposed by [78], a simple range query (i.e. a square window query) is performed. That way, a larger area around the citizen is considered (approximately 21.46% wider), but the query is kept simple and its execution time is lower. In fact, no distance function inside the query itself is computed at all. Instead, the area is specified through two `BETWEEN` statements in the `SQL WHERE` clause.

Objects falling into the exceeding area will be discarded by the Android client after an exact evaluation of their relative distance from the user. The client itself will also be responsible of bounding box evaluation and objects sorting. These concepts are shown in Figure 2.5. The offloading of part of the computation from the DB server to the mobile client let us reduce the workload of the DBMS, while maintaining a precise calculation of the desired area. The improvement achieved in the execution time of the query is on average 23.5%. Scaling up the time of a single execution to a

number of queries that can reach thousands per minute, the benefits of this approach are evident.

The bounding box is computed using the Java GeoLocation class provided in [78], in order to avoid possible miscalculations in the regions near the North and South geographic poles and surrounding the 180th Meridian.

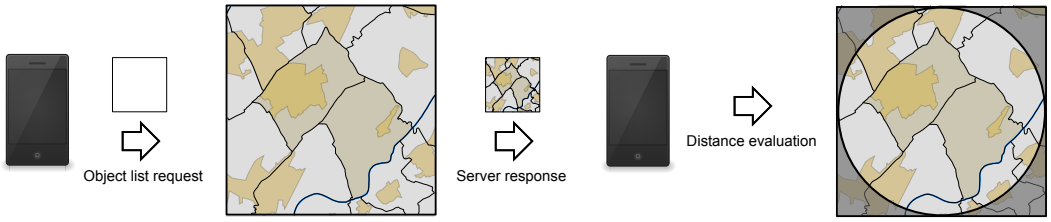


Figure 2.5: The logic behind an object list remote query: a squared bounding box on the Android client is computed, then the database server performs a simple range query on that box. Finally the Android client is responsible of tasks like object sorting and real distance evaluation.

2.0.7.6 Indices

A requisite of primary importance for a responsive OLTP architecture is a fast execution of SQL queries. It was already described how pre-processing and post-processing performed on the mobile client can ensure a lighter computational load on the server and a simpler query syntax. However, the most important implementation choice is certainly the kind of indices for database tables. In the described context the database will be continuously queried for objects in small square regions. This kind of query, called *range query*, is notoriously well served by B+-trees [28] or, in the case of a geometrical context, by R-trees [42]. As we consider geo-located objects, both R-trees and B+-trees are tested, in order to select the most performing option for the application. Since MySQL does not natively support the more advanced kNR-trees [82], in the first case a simple R-tree is considered, and access specific categories using the SQL `WHERE` clause.

In the proposed tests both stand alone and distributed MySQL servers are considered. The most computationally intensive queries are performed on the `object` table. In particular, spatial information needs to be indexed: the `longitude` and `latitude` fields, for B+-trees, as well as the `point` field in the case of R-trees. The `point` field is a MySQL representation of a geometric point (containing one longitude and one latitude value), belonging to the `geometry` class. In fact, MySQL supports R-tree indices only for `geometry` data.

MySQL uses R-Trees with quadratic splitting for `SPATIAL` indices on spatial columns. A `SPATIAL` index is built using the *Minimum Bounding Rectangle* (MBR) of a geometry [132]. For most geometries, the MBR is the smallest rectangle that surrounds the geometry. For a point, the MBR is a rectangle degenerated into the point. For tables stored with MyISAM storage engine, `SPATIAL INDEX` creates an R-tree index. It is also possible to index a spatial column with a B-tree. Nevertheless, a B-tree index on spatial values will be useful for exact-value lookups, but not for range scans. That is why a B+-trees is built upon `longitude` and `latitude` fields (that are stored in `float(10,6)` format) and a R-tree is built upon `point` field (that is stored in a MySQL `POINT` data structure). The integrated query optimizer uses available spatial indices every time functions such as `MBRContains()` or `MBRWithin()` are present in the `WHERE` clause of the query.

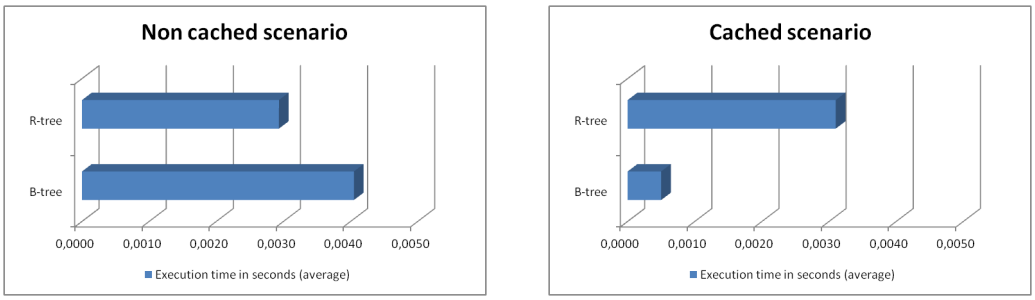
The two following SQL queries reflect the statements that were actually used in the experiment, the first one for B+-tree indices and the second one for R-tree indices. These two queries are syntactically different but they have the same semantic meaning, i.e. they query the data base for objects in the same bounding box.

```
SELECT id, name, idCategory, longitude, latitude FROM object
WHERE latitude BETWEEN x1 AND x2 AND longitude BETWEEN y1 AND y2;
```

```
SET @boundingBox = 'Polygon((x1 y1, x2 y1, x2 y2, x1 y2, x1 y1))';
SELECT id, name, idCategory, AsText(point) FROM object
WHERE MBRContains(GeomFromText(@boundingBox),point);
```

For each kind of index, multiple tests have been carried out, each selecting areas with a different number of objects. The experiment was further expanded first by evaluating additional queries for which a single object category is retrieved, and then by enabling and disabling the caching of the queries.

Query caching is an important factor for the performances of an OLTP architecture. Test results on a realistic instance of a sample geographical dataset in Figure 2.6 show a remarkable difference between cached and non cached queries. In the non cached scenario, R-trees are approximately 30% faster than B+-trees, while in the cached scenario B+-trees are roughly 84% faster than R-trees. These tests were performed on a single server. In the distributed scenario considered in Section 2.0.7.2 the MyISAM storage engine can not be used, since a MySQL cluster requires the



(a) (b)

Figure 2.6: The average plotted results of indices testing. Cached and non-cached scenarios are considered separately. R-trees perform better in a non-cached context while B+-trees do in the cached one.

NDBCluster storage engine. Since NDBCluster does not support `SPATIAL` indices, in a distributed context only B+-trees can be adopted.

It is remarkable that the best way to obtain a scalable and high performance OLTP architecture consists in using NDBCluster tables indexed with B+-trees and a wide usage of query cache. However, a query can be recalled by MySQL from the cache only if it is exactly identical to the current one. The comparison is a real byte to byte equality check, so, for instance, `SELECT * from object` is different from `select * from object`. Since even two client devices located only meters one from another would produce queries with different longitude and latitude parameters, query caching can not be effectively used. A solution to this limitation could be to pre-process the bounding box on the server and then match the queries to already cached queries from devices within a certain distance, using for example some hash criteria. These issues surely represent an interesting subject for future research.

2.0.8 Security of a location-aware application

Security is a crucial factor for the success of a complex infrastructure like the one just discussed. Moreover, wherever user interaction is required, and user accounts are created, privacy measures need to be taken, according to current legislation.

User authentication is one of the most critical components from a security point of view. Password are transmitted through cryptographic techniques, and HTTP authentication mechanism is designed with SQL injection resistance in mind. Direct SQL Command Injection is a technique where an attacker tries to alter existing SQL commands to modify or get access to hidden data. If no precautions are taken, an attacker can even execute dangerous system level commands on the database host.

This is accomplished through the use of badly formatted user input given to the application. This malicious inputs usually use static parameters to build an SQL query.

As suggested in [80], a form of stack guard that prevents the database from being SQL injected is adopted. First of all, every parameter read from the HTTP POST request (shown in table 2.2) is truncated to the maximum sensible length and passed to a function that escapes special characters, taking into account the current character set of the data base connection so that it is safe to be placed in a query. Moreover, each query executed through PHP within the server HTTP interface (2.0.7.4) runs under the MySQL Improved Extension (mysqli). Queries are only performed through prepared statements, following the *prepare - bind - execute* pattern; that way a statement template is send to the database server, the server performs a syntax check and initializes internal resources for later use. The client can bind parameter values and sends them to the server. The server creates a statement from the statement template and the bound values to execute it using the previously created internal resources.

For authentication purposes, only the hash of the password is stored and transmitted. The chosen hash function is SHA-512, part of the SHA-2 family, and currently the strongest function approved by the United States National Institute of Standards and Technology (NIST) [49].

As proposed in [29], additional security checks should be performed when sensitive data are stored in the cloud. In particular, Curino et al. propose the use of an enhanced security scheme that allows SQL queries to be run over encrypted data, including ordering operations, aggregates and joins. As cloud services are expected to sustain a strong growth in the near future, and are candidate to partially replace in-house computing solutions, the issue surely represent an interesting research topic for future work.

2.0.9 Indoor positioning

The emergence of smarter environments in urban areas, along with the improvements achieved in the context of global satellite positioning systems, led to an intensive use of outdoor location-aware applications [23]. Meanwhile, the scientific and industrial community realized the potential and benefits of location-aware applications in the indoor environment [79]. In the last decade the need to track persons and objects in indoor environments has become increasingly important for several reasons.

Nowadays, indoor positioning is applied in several fields: customer navigation in a mall, citizen navigation inside a public building, product localization in a supermarket

and indoor location-aware advertisement are just a few examples. Unfortunately, while the performance of outdoor positioning systems has become excellent, indoor positioning seems to be more complicated [100]. Apparently a single multipurpose solution that fits any need in indoor positioning applications is still missing. In fact, the systems proposed in literature rely on several technologies and tailored models designed in order to fit the requirements of each specific context [71].

Since 2007, the European Union defined several factors that should be considered when evaluating the *smartness* of a city [39]. Healthcare plays a key-role when the quality of life concerning a urban area is to be estimated [111]. The availability of modern and citizen-friendly hospitals is one of the factors that belong to the *Smart Living* macro area: [46] state that telemedicine, electronic records, and health information exchanges in remote assistance are meaningful case studies on the subject. The possibility to track patients while they are inside a hospital and especially in first aid area is becoming increasingly important. In these areas emergencies frequently occur and this causes a constant flow of doctors and nurses from one room to another. This to-and-fro together with specific mental disease (e.g., Alzheimer disease), often leads to lost or forgotten patients around the hospital. Hence, knowing the right position of a person inside a medical facility at any time is a meaningful problem within indoor localization research field. Unfortunately, the medical scenario is a particularly hostile context for indoor localization because the transmitters and receivers usually adopted to locate objects have to deal with medical devices; indeed these machineries often come with some limitation on the allowed radio-frequency range to be used near them and so on. For these reasons, the author investigated the literature and proposed a novel technique for tracking patients inside a hospital [21].

In the following a focus on indoor positioning applied to hospital and medical scenarios is posed; an expert system capable of locating people within a hospital emergency unit is presented. This system adopts RFID transmitters and receivers and handles their signals through a dedicated infrastructure. Patients are located through a classification of these signals via a hierarchical *Random Forest*-based classifier proposed herein.

In recent years several system models concerning indoor positioning were presented. For the purposes of this thesis it is meaningful to focus on the system models for indoor localization based on classifiers, as the herein proposed method lies in this field.

[48] provide a detailed performance analysis of a k-Nearest Neighbour based positioning system and also propose a positioning scheme based on Gaussian Mixture

Models in order to locate objects in a generic building.

An indoor positioning technique relying on bayesian classifiers focusing on *smart office* and *smart building* scenarios is presented by [61].

A valuable study on the subject was proposed by [129]; it consists of a practical experiment which took place inside a University. Here a Wi-Fi based system is used to gather signals and a wide range of classifiers are compared during the localization process. [116] suggest that a viable method to improve the classification process in the Wi-Fi indoor positioning context is to combine classifiers, and provide subsequently some practical case studies on the subject [117].

While [65] provide a general survey on indoor positioning methods, [41] focus on positioning systems for wireless personal networks.

In another key-survey [71], the authors describe a wide range of indoor localization systems and compare them on the basis of some performance metrics (*Accuracy*, *Precision*, *Complexity*, *Robustness*, *Scalability*, *Cost*). In the following a reference to some of these key-features is posed, in order to provide a helpful comparison with other systems. The survey also describes several algorithm and mathematical techniques used for indoor positioning purposes.

Mathematical approaches for indoor localization are however better described by [115] and divided in four main categories: *geometry-based methods*, *cost function minimization methods*, *fingerprint methods* and *bayesian methods*. According to this taxonomy, it is reasonable to include the proposed system among the fingerprint based ones. Fingerprint methods rely on two main steps. In the first one, the signal levels from the different base stations are recorded in order to form a training set and calibrate the system. In the localization step, the system tries to locate the source of a new signal classifying it against the previously recorded set. These methods are particularly robust in the *non-line-of-sight* (NLOS) context, where radio transmission across a path is partially obstructed, usually by a physical object.

Most of the ICT works in the *Smart Health* field relate to the design of proper information systems for medical purposes or to the innovative techniques adopted to cope with several known diseases [136]. On the contrary, this thesis describes a novel technique to track patients in an emergency room environment in order to find them quickly as needed. Specifically, in this context, it is important to know the room where the patient is located rather than his precise position. The system proposed herein is aimed to locate patients through RFID signals classification. In particular, signals are processed combining several instances of *decision-tree* classifiers called Random Forest [16]. As assessed by [134] the off-line phase (training) of a fingerprint

method is not time critical whereas on-line phase (localization) is really time critical. Decision-tree oriented classifiers behave well according to this time constraint.

2.0.10 Indoor localization in a hospital environment

In general, it is quite complicated to obtain satisfying results concerning indoor positioning inside a hospital or a medical facility using wireless techniques. In this specific scenario, several problems may occur due to location sensors installation. Frequently, medical devices conflict with installed sensors; moreover, these buildings are often equipped with shielded walls. For instance, we could imagine an x-ray examination room equipped with thick leaded walls [45]. This leads to a tangible signal reduction or distortion during the communication between transmitters and receivers.

The deployed infrastructure relies on *Radio Frequency Identification* (RFID) technology and consists of three different sensors. Each sensor communicate in *Ultra High Frequency* (UHF) mode on frequencies *LPD 433 MHz*, *446 MHz* and *860 MHz*. Keeping the frequency range as confined as possible allows the infrastructure not to conflict with other WiFi and medical devices.

Patients are equipped with an active RFID transmitter (RFID Tag). Each *Tag* lies inside a small hypoallergenic bracelet and sends a signal on an user-defined time interval basis. The duration of each signal is limited to a few milliseconds, resulting in a total effective transmission time of few hours per year. The irradiance at a point of the surface is near to 0. One of the main concerns in dealing with active RFID tags is that of batteries and power consumption. Active tags send signals to the receiver on their own and thus need a local power source. As signals related to the adopted tag are really short and energy-preserving, bracelets can be equipped with a small battery which can although keep the tag alive for more than three months.

Signals sent from tags are received from several antennas (RFID Receiver) which store the Tag Identifier (ID) and the strength of incoming signal. Specifically, signal strength is expressed in decibels and varies from $-100db$ (weak signal) to $-30db$ (strong signal). Receivers stay idle until a signal from a tag occurs and, after its processing, which lasts for a few milliseconds, go back to sleep mode. Thus, these antennas are energy-preserving as well and can be both plugged to the hospital power supply or equipped with batteries.

Each antenna sends the data to a central receiver (on the same frequency range) which is connected via an *RS232-to-Ethernet* adapter to the local server. This central receiver collects the signals, stores them and performs real-time localization.

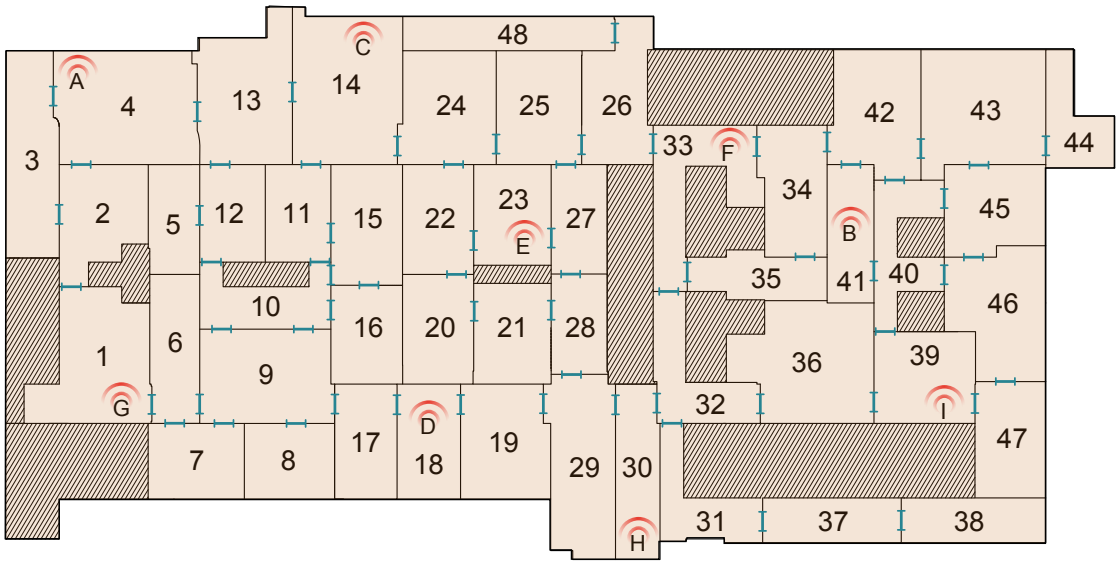


Figure 2.7: The detailed map of the hospital's emergency unit where the tests were made. Rooms are labeled with numbers (from 1 to 48) while the deployed antennas are labeled with letters (from 'A' to 'I'). Openings between rooms are highlighted as well.

These features make the infrastructure compliant with *Industrial, Scientific and Medical* (ISM) radio frequency specifications and suitable for the hospital environment [52].

The emergency unit where the proposed system has been deployed is divided in 48 rooms covering an area of about $4000m^2$. After the acceptance procedure has been carried out, each patient is routed to a specific area, depending on his personal disease. Following the opinion of the hospital authorized personnel, 9 suitable locations have been determined to install 9 antennas across the unit. Specifically, each antenna was positioned following two criteria: to have the emergency unit fully covered and to allow signals generated from each room to be received by at least three antennas. Some of these antennas are not able to receive signals while transmitters are located in certain rooms. This behavior is sometimes due to signal reduction or distortion or to the distance between transmitter and receiver, as the emergency unit is quite wide (see Figure 2.7). An overall representation of the emergency unit is provided in Figure 2.7.

2.0.10.1 Data acquisition

A dataset has been acquired through receivers and transmitters described in Section 2.0.10 by a team of experts, trying to simulate real operating conditions. To this pur-

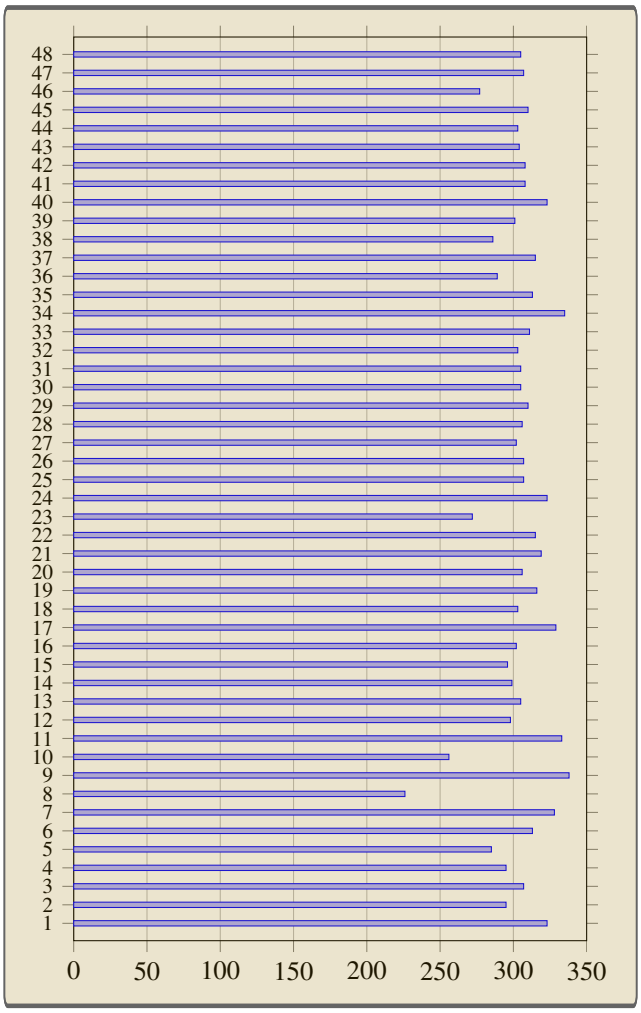


Figure 2.8: Number of observations registered for each room.

pose, the acquisition was performed while the emergency unit was fully operational. Thus, these data are suited to face the small changes which often take place in such a context. In particular, four different individuals wearing active RFID bracelets, moved through each room of the building. Bracelets sent a signal to the antennas on a five seconds basis. Data acquisition lasted for about five consecutive hours, resulting in 14622 observations from the 48 rooms. Figure 2.8 shows the distribution of these observations among each room. On the average, there are 304.6 observations per room. Rooms 8 and 9 hold the minimum and the maximum number of observations, respectively (precisely 226 and 338).

Figure 2.9 reports, for each receiver, the percentage of observations detected. According to the chart, it is clear that C and D are the antennas with the highest reception rates (52.3% and 61.5%, respectively), while B, G, and I are those with the lowest ones (36.5%, 36.1% and 24.6%, respectively).

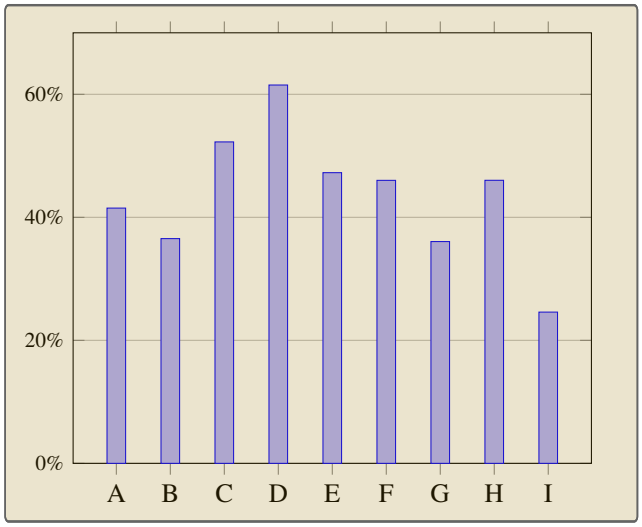


Figure 2.9: Percentage of valid observations received by each antenna.

Definition 1 *Given a transmitted signal, it is considered to be a valid value with respect to an antenna* A *if and only if* A *is able to receive it.*

Definition 2 *Given a set of d antennas and an integer w, $1 \leq w \leq d$, a valid observation is an observation holding at least w valid values.*

Figure 2.10 shows the percentage of observations with at least w valid values ($1 \leq w \leq 9$): each observation presents 3.9 valid values on the average. Apparently, when $w \geq 4$, the percentage of discarded observations (i.e., observations with a number of valid values less than w) becomes too high (i.e., from 38.9% to 99.9%), undermining the system usability.

Figure 2.11 reports, for each receiver, the range of signals' strength through *box-plots*, which provide detailed statistics for each antenna. This chart shows clearly that all antennas receive signals within the range $[-100, -70]$ in 75% of cases, resulting in a reduced signals' strength range.

Overall, some conclusions can be drawn:

- The dataset is very difficult to handle due to the high percentage of missing values (see Figure 2.9 and Figure 2.10);
- The localization task is rather difficult since, in several cases, signals received from different rooms are very similar (e.g., rooms 40, 41 and 42 holds signatures close to each other).

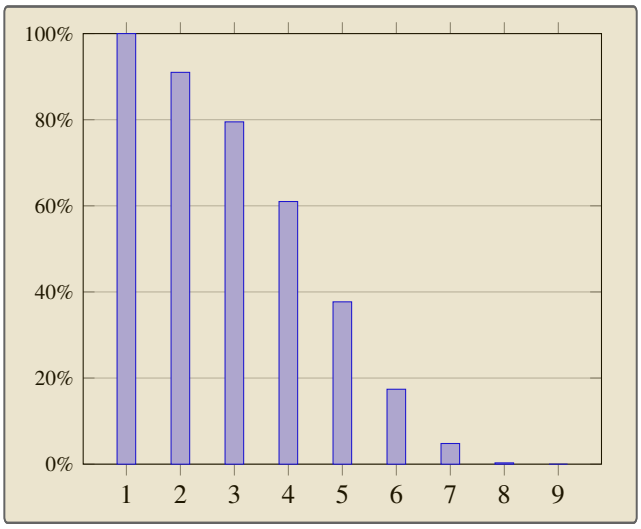


Figure 2.10: Percentage of observations presenting at least w valid values; the result is reported for increasing values of w.

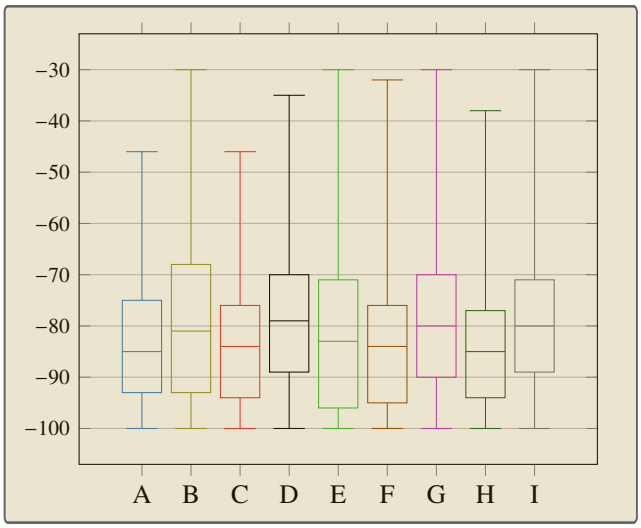


Figure 2.11: Boxplot on the range of values observed by different antennas. Specifically, for each receiver the median, minimum, maximum, 25th and 75th percentile values are given.

- A future study on the rearrangement of certain antennas could improve the system performance with respect to its robustness. For instance, B, G and I could be moved in an area featuring higher reception rates (see Figure 2.9);

2.0.10.2 Localization system

The idea behind the proposed system is to organize rooms in a hierarchical structure through a partitioning into non-overlapping macro-regions. Each macro-region contains rooms with similar observations and is assigned separately to a specific sub-

system, customized to manage the data related to that macro-region.

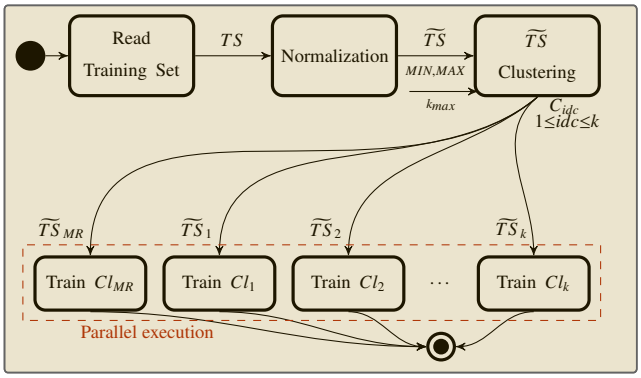


Figure 2.12: Training process.

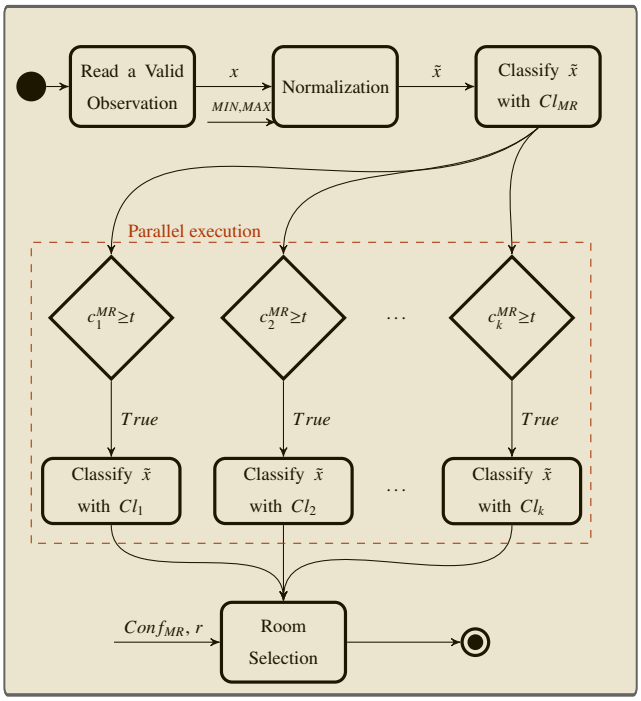


Figure 2.13: Localization process.

Figures 2.12 and Figure 2.13 show the learning and localization activity diagrams of the proposed system, respectively. In the following, specific steps involved in training and localization procedures are described in detail.

Let $TS = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^n\}$ be a set of d-dimensional valid observations, where each element $x^i_j, j = 1, \ldots, d$ represents the value received by the antenna j for the observation $\boldsymbol{x}^i$. A class label is associated to each observation and it represents the room identifier related to that acquisition. The training set TS is used to train the localization system. This procedure could be time consuming but usually it is

performed only once during the system setup. The training stage can be divided into three steps:

- Data normalization: pre-processing stage aimed at extracting a feature vector from each observation;
- Macro-regions definition: room partitioning into non-overlapping regions characterized by similar observations;
- Classifier training: training of region-specific classifiers.

The normalized training set $\widetilde{TS} = \{\tilde{\boldsymbol{x}}^1, \tilde{\boldsymbol{x}}^2, \ldots, \tilde{\boldsymbol{x}}^n\}$ is obtained by mapping each training pattern in the range $[0; 1]$. This normalization is aimed at representing the data in a easier-to-manage format. The acquired values are in fact represented by negative integers in the range $[-100; -30]$; moreover, as shown in Section 2.0.10.1, observations present a considerable number of missing values ($x_j^i = 0$) which are, in this phase, replaced by the fixed constant -1.

The first step of data normalization consists in determining, for each receiver, the minimum and maximum value in the training set, so that all the values can successively be normalized in the range $[0; 1]$ accordingly. Let us define $MAX = \{M_1, M_2, \ldots, M_d\}$ and $MIN = \{m_1, m_2, \ldots, m_d\}$ as the sets of maximum and minimum values observed for each receiver over the whole training set TS respectively. Specifically

$$M_j = \max_{\boldsymbol{x}^i \in TS, x_j^i \neq 0} \left(x_j^i\right), \tag{2.1}$$

$$m_j = \min_{\boldsymbol{x}^i \in TS, x_j^i \neq 0} \left(x_j^i\right). \tag{2.2}$$

For each $\boldsymbol{x}^i \in TS$, the normalized value of each single feature $\tilde{x}_j^i, j = 1, \ldots, d$ is calculated as follows:

$$\tilde{x}_j^i = \begin{cases} \frac{x_j^i - m_j}{M_j - m_j}, & if x_j^i \neq 0 \\ -1, & o.w. \end{cases} \tag{2.3}$$

Clearly, normalization helps to spread different observations and thus it simplifies the classification task.

Different rooms, each represented by a set of feature vectors extracted according to the normalization procedure, are then grouped into non-overlapping regions. In

the following, depending on the context, we will refer to clusters or macro-regions interchangeably. The partitioning algorithm operates on $\widetilde{TS}$ and is based on a modified version of the *k-Means* algorithm [75, 33], which supports missing values. The modified algorithm is composed of the following steps:

1 Set $k = k_{max}$ random points into the d-dimensional space $[0; 1]^d$. These points represent initial cluster centroids.

2 Assign each room to the cluster holding the centroid closest to most of the observations belonging to the room. Since most of those observations contain missing values, the *Partial Distance* (PD) [33] instead of the common *Euclidean Distance* is used to calculate the distance between two points. PD is calculated as follows:

$$PD\left(\boldsymbol{x}, \boldsymbol{y}\right) = \frac{d}{d - \sum_{j=1}^{d} b_j} \cdot \sum_{j=1,\ldots,d \wedge b_j = 0} \left(x_j - y_j\right)^2, \tag{2.4}$$

where x_j and y_j represent the j-th values of the d-dimensional points $\boldsymbol{x}$ and $\boldsymbol{y}$, and

$$b_j = \begin{cases} 0, & if x_j \neq -1 \wedge y_j \neq -1 \\ 1, & o.w. \end{cases} \tag{2.5}$$

PD computes the squared Euclidean Distance between all valid values and normalizes it by the reciprocal of the proportion of valid values used during the calculation.

3 When all rooms have been assigned, recalculate the position of the k centroids. If there are no rooms associated to one or more clusters, delete them and update the number of valid clusters k. Let idc be the identifier of a cluster, let $\widetilde{TS}_{idc}$ be the set of normalized observations belonging to rooms assigned to this cluster and let $\boldsymbol{\mu}^{idc}$ be the cluster's centroid. The coordinates of each centroid are calculated as the arithmetic mean of all valid observation values of the rooms assigned to the corresponding cluster:

$$\mu_j^{idc} = \frac{1}{\sum_{\boldsymbol{x} \in \widetilde{TS}_{idc}} v_j} \cdot \sum_{\boldsymbol{x} \in \widetilde{TS}_{idc},\, v_j = 1} x_j, \tag{2.6}$$

where $1 \leq idc \leq k$ and $1 \leq j \leq d$, having

$$v_j = \begin{cases} 1, & if x_j \neq -1 \\ 0, & o.w. \end{cases} \tag{2.7}$$

4 Repeat steps 2 and 3 until no modifications to the centroids are performed or a maximum number of iterations is reached.

Figure 2.14 depicts an example of room grouping into five macro-regions. Note that, since the clustering is performed according to radio signal features, which carry no spatial information, macro-regions may also contain non-adjacent rooms (e.g., rooms 7 and 42).

Once the rooms are grouped into k macro-regions, different classifiers are trained using different training sets derived from $\widetilde{TS}$. Note that, this operation can be performed in parallel since there are no dependencies between different classifiers and training sets. A macro-region classifier Cl_{MR} is trained using $\widetilde{TS}_{MR}$, a dataset containing all the data in $\widetilde{TS}$ where the identifier of each room has been replaced by the corresponding macro-region identifier. Moreover, for each cluster, a classifier Cl_{idc} is trained using $\widetilde{TS}_{idc}$, a $\widetilde{TS}$ subset containing only the observations from rooms belonging to cluster idc. Due to the high number of rooms, the chosen hierarchical organization allows each sub-classifier Cl_{idc} to focus on a specific and smaller room subset, thus making easier the classification problem.

Among the classifiers suitable for the algorithm implementation, the *Random Forest* [16] has been selected. This choice was due to the classifier convenient behavior both in terms of accuracy and efficiency, which are critical features of such an application. In fact, the training process of the whole proposed system requires about 2 seconds on the average. The localization task is instead performed in real-time and its related execution time is negligible. These values are referred to an Intel Core 2 Quad Q9400 CPU at 2.66 Ghz, using a multithreaded C# implementation. Random Forest is an ensemble classifier that consists of a set of decision trees. Let N be

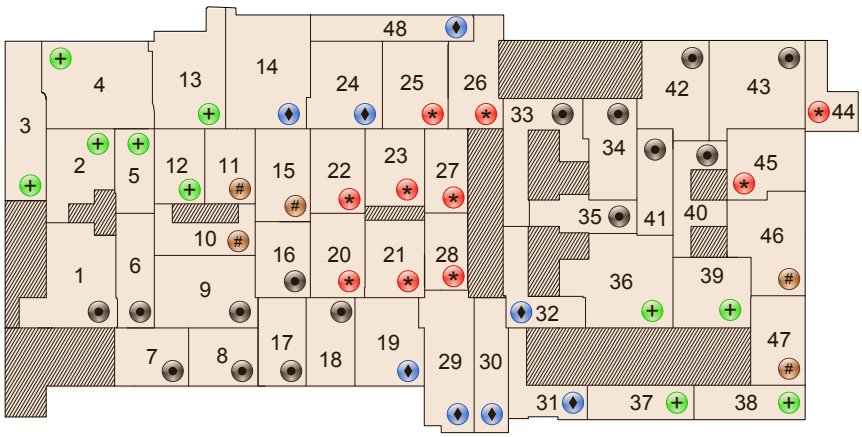


Figure 2.14: An example of room grouping. Here five clusters are generated.

the number of samples in the training set and P be the number of variables in the classifier; let p be the input parameter denoting the number of input variables to be used to determine the decision at a node of the tree (usually $p \ll P$). Each tree is grown as follows:

- Take a bootstrap sample by choosing n times with replacement from all N available training samples;
- Use the remaining samples to estimate the error of the tree, by predicting their classes;
- For each node of the tree, randomly choose p variables on which to base the decision at that node. Calculate the best split based on these p variables in the training set;
- Each tree is fully grown and not pruned.

To classify a new object, the sample is given as input at each of the trees in the forest; each tree gives a classification and the final class is chosen on the basis of the majority vote rule.

On the basis of the distribution of votes among the different classes, a confidence is also computed for each class. Specifically, given a class i, the related confidence c_i corresponds to the percentage of votes obtained through the set of trees composing the classifier.

During the localization, a new observation $\boldsymbol{x}$ is given as input to the system in order to predict the room where the subject currently resides. The localization procedure can be divided into three steps:

- Data normalization;
- Macro-region classification;
- Room classification.

The new observation $\boldsymbol{x}$ is first normalized as described hereabove, obtaining $\tilde{\boldsymbol{x}}$. Since some feature values could be out of the maximum and minimum bounds determined in the training phase, a clipping operation is performed to preserve values in the range $[0; 1]$:

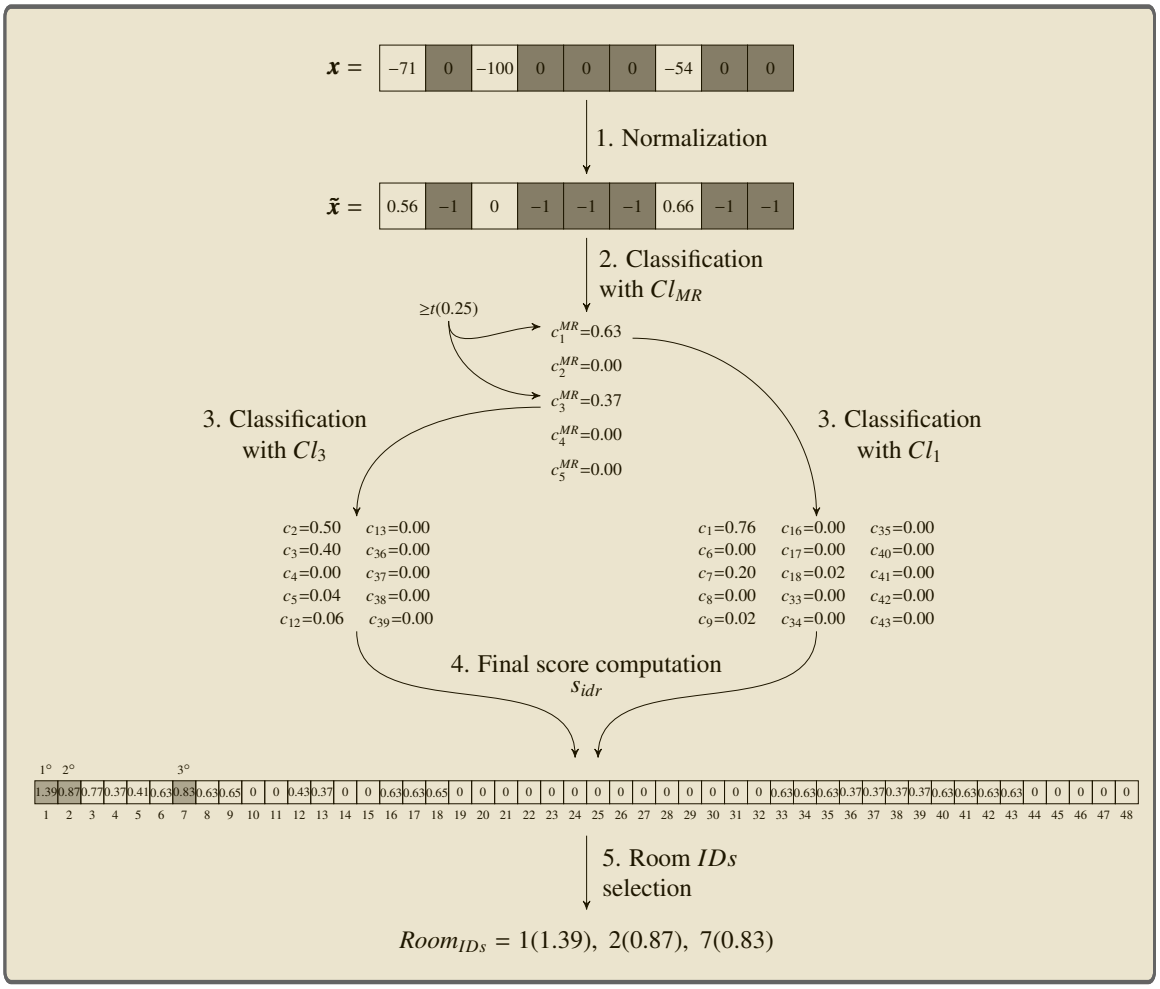


Figure 2.15: Example of localization process, given a valid observation. The room grouping proposed in Figure 2.14 is used. Each c_i represents the confidence that the observation come from room i.

$$\tilde{x}_j^i = \begin{cases} 0, & v < 0 \\ 1, & v > 1 \\ v, & o.w. \end{cases} \tag{2.8}$$

where

$$v = \frac{x_j^i - m_j}{M_j - m_j}. \tag{2.9}$$

The classifier Cl_{MR} is used to pre-select those macro-regions where, according to a given observation, the probability of presence of the subject is higher than a prefixed threshold. The normalized observation $\tilde{\boldsymbol{x}}$ is used as input for the classifier Cl_{MR} that returns a set of confidences $Conf_{MR} = \{c_{idc}^{MR}, 1 \leq idc \leq k\}$ where c_{idc}^{MR} represents the likelihood that the subject is located in a room belonging to the macro-region idc.

The classifiers corresponding to pre-selected macro-regions, are used to predict the rooms from which the signal could have been generated. The normalized observation $\tilde{\boldsymbol{x}}$ is given as input to each classifier Cl_{idc} for which the corresponding macro-region confidence c_{idc}^{MR} is greater than or equal to a predefined threshold t. The result is a set of confidences $Conf_{idc} = \{c_{idr}, idr \in C_{idc}\}$ where C_{idc} is the set of all identifiers of rooms assigned to the macro-region idc and c_{idr} represents the likelihood that the subject is located in room idr. Note that, this operation can be performed in parallel since there are no dependencies between the different classifiers.

The final score of each room is calculated as:

$$s_{idr} = c_{M(idr)}^{MR} + c_{idr}, \tag{2.10}$$

$$M\left(idr\right) = idc,\ idr \in C_{idc}, \tag{2.11}$$

where $M\left(idr\right)$ returns the identifier of the macro-region containing the room idr. In particular, the confidence of the macro-region classifier is combined to that of the selected classifiers on the basis of the sum rule. Finally, the identifiers of the r rooms with the highest scores are returned. Note that, in the proposed Random Forest-based implementation, both confidences $c_{M(idr)}^{MR}$ and c_{idr} are computed as described before. The first one is provided by the classifier Cl_{MR}, trained to distinguish among the different macro-regions while the second one is returned by the corresponding room classifier Cl_{idc}.

Figure 2.15 shows the whole localization procedure over a sample observation, using room grouping as reported in Figure 2.14.

2.0.10.3 Experimental evaluation

This section describes the experiments carried out to evaluate the performance of the proposed system in terms of accuracy, precision, complexity, robustness and scalability [71].

The localization system must be trained before it can be used to localize the subjects. To this purpose, the dataset described in Section 2.0.10.1 has been divided into two disjoint subsets: half of the observations belonging to each room is randomly selected as training data while the other half is used as test data.

To assess the impact of invalid values on the system performance, experiments have been carried out discarding (both in training and test sets) those observations that present a number of valid values less than w, with $w = 1, 2, 3$. Moreover, the localization system has been evaluated for different values of the parameter $r = 1, 2, 3$,

representing the number of rooms returned by the system. Finally, to reduce performance indicators variability the experiments have been repeated 20 times, each time randomly choosing complementary training and test sets, and the final performance indicators have been calculated as results' average.

Accuracy Generally, accuracy is measured as the average distance between the estimated location and the real location. In this specific context, given two room identifiers i and j, the distance $d(i,j)$ between the rooms is recursively calculated as follows:

$$d(i,j) = \begin{cases} 0, & i = j \\ \min\limits_{k \in A(i)} d(k,j) + 1, & o.w. \end{cases} \tag{2.12}$$

where $A(i)$ contains the identifiers of all rooms adjacent to room i. Two rooms are adjacent if they share a wall or at least a part of it (e.g., in Figure 2.7, rooms 1, 3, 4 and 5 are adjacent to room 2). The distance d represents the minimum number of walls that separate the true room and the estimated one (e.g., $d(4,7) = 3$). In other words, a distance of zero means the system has located the right room, a distance of one means the real room and the located one are adjacent and, more in general, a distance of u means that between the right room and the located one there are u walls.

Please note that the definition of accuracy used by [71] is here applied. This definition differs from the concept of accuracy usually adopted in other contexts, since the lower is the value (lower distance), the more accurate is intended to be the classification system. Table 2.4 reports the average accuracy of the proposed system and the corresponding standard deviation for different values of r and w, representing respectively the number of rooms retrieved by the localization system and the minimum number of valid values used to select valid observations. As proved by standard deviation values, accuracy's variability observed after the 20 tests have

r / w	**1**	**2**	**3**
1	0.76 +/- 0.02	0.55 +/- 0.01	0.47 +/- 0.01
2	0.40 +/- 0.01	0.25 +/- 0.01	0.20 +/- 0.01
3	0.27 +/- 0.01	0.16 +/- 0.01	0.12 +/- 0.01

Table 2.4: Accuracy of the proposed localization system as a function of the number of rooms retrieved (r) and the minimum number of valid values used to select the observations (w).

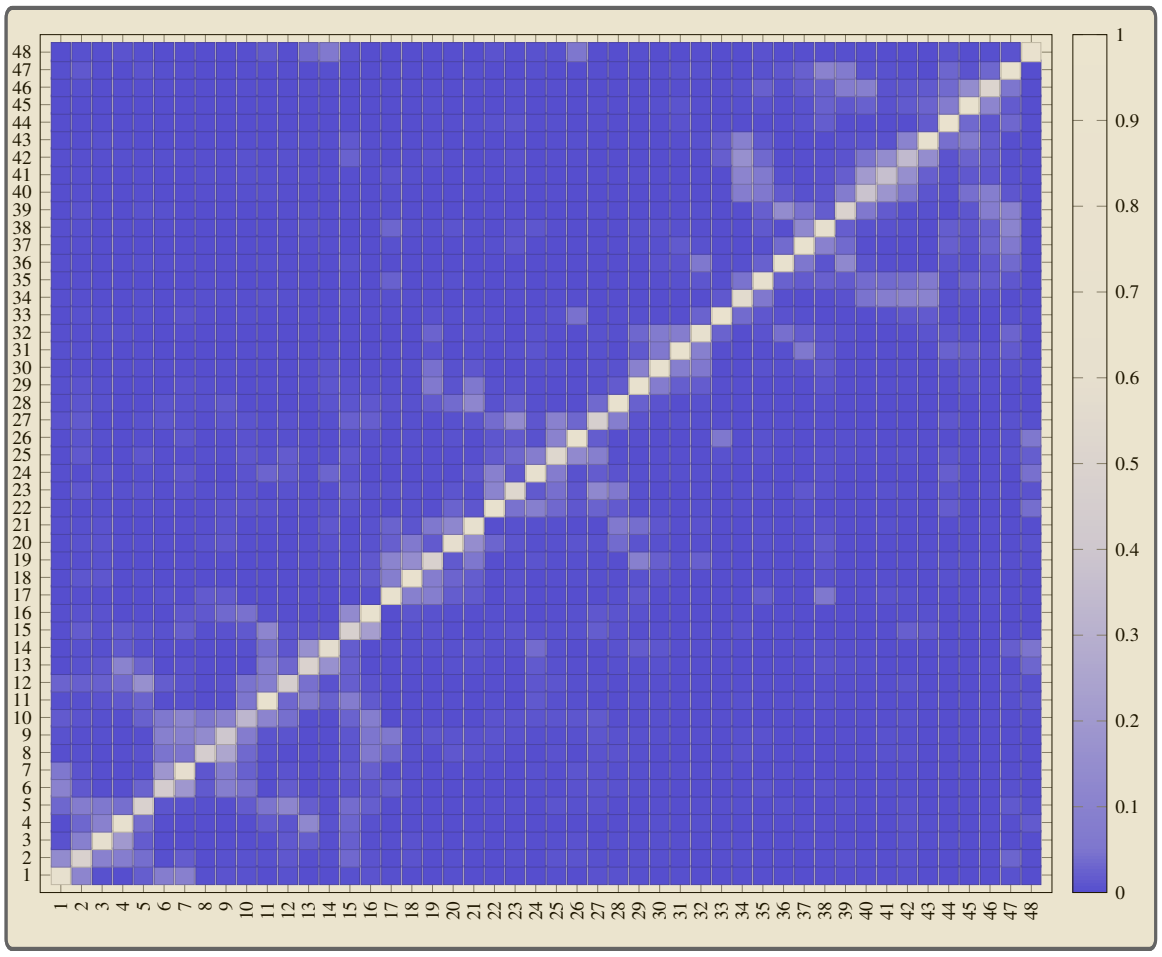


Figure 2.16: Confusion matrix. Blue values indicates a low percentage of localization while lighter ones indicates a higher localization rate.

been carried out is very low. Note that, when $r > 1$ the distance used to calculate the accuracy is the lowest one among the r distances calculated.

This chart shows that, as expected, better results in terms of accuracy can be reached by applying more restrictive constraints on the number of valid features required ($w = 2$ or $w = 3$); it is worth noting that the limits imposed in these experiments are rather low with respect to the number of receivers available. Similarly, accuracy improves when a higher number of rooms ($r = 2$ or $r = 3$) is retrieved by the localization system. According to the constraints applied to the problem discussed herein, a value of $r = 2$ is acceptable; overall, the results provided using $r = 2$, $w = 2$ and $r = 2$, $w = 3$ are both satisfactory in terms of accuracy and suitable for a practical implementation.

Figure 2.16 depicts the confusion matrix where each row represents a room identifier (expected for classification) and each column states the resulting localization. In other words, element (i, j) represents the percentage of signals transmitted from

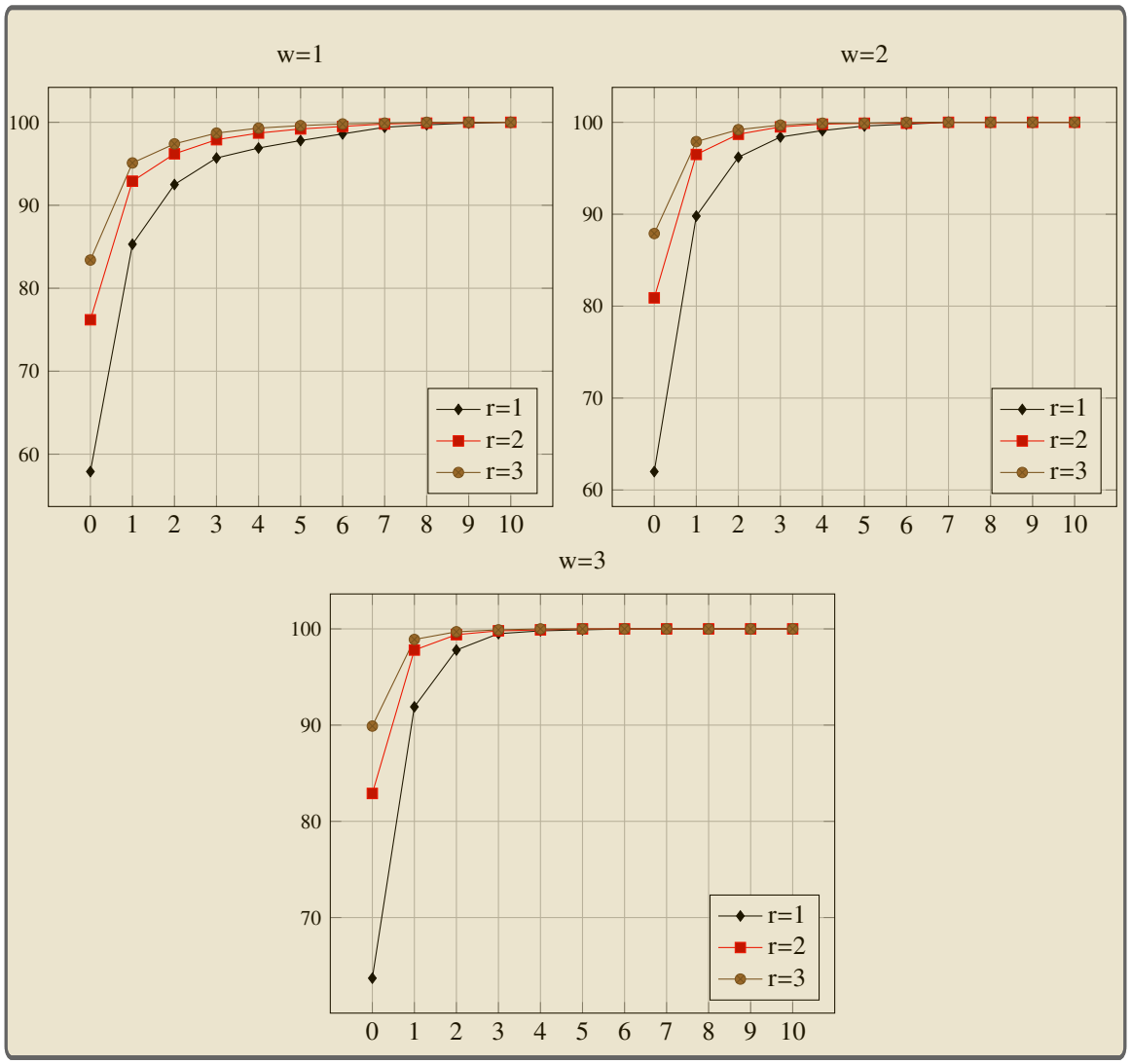


Figure 2.17: Cumulative distribution functions of the distance error, reported for different values of r and w.

room i localized as deriving from room j. This matrix helps to understand whether or not the system is confusing two rooms during localization. As predictable, high values are mainly distributed along the diagonal; it is thus confirmed that the system holds a high accuracy level. This matrix also confirms the frequent localization errors related to rooms 40, 41 and 42, previously explained in Section 2.0.10.1, as the area around those room identifiers presents a notable blur.

Precision Usually, the precision of a localization systems is defined as the distribution of distance error between the estimated location and the real location. The precision of the proposed system is measured using the *Cumulative Distribution Function* (CDF) of the distance error. In particular, charts reported in Figure 2.17 show

CDFs of the proposed system for different values of r and w. Each chart represents the percentage of correctly classified patterns as a function of the distance defined in (2.12)). According to this test, the behavior observed in the previous experiment is confirmed. In particular, by selecting only observations with a minimum of two valid values ($w = 2$), localization is performed with a maximum distance of 1 in more than the 90% of the tests that have been carried out (with respect to the real room). So, the localization system tends to identify either the correct room or one of its adjacency.

Complexity Complexity of a localization system can be attributed to hardware, software and operation factors. For simplicity, it is usually estimated as the computing time required to perform a localization. As reported hereabove, the system performs a localization in a negligible time, and it thus can be adopted in real-time applications.

Robustness In general, robustness measures the capability of a localization system to work normally even if some signals are not available due to receiver errors. The results in terms of precision (see Figure 2.17) show that the proposed localization system holds good robustness properties. The proposed algorithm was designed in order to operate with many missing values, allowing to reach satisfactory results even with a limited number of valid features available (for instance, an acceptable parameter could be $w = 2$, meaning that only two valid values, among 9, are required in order to accept an observation as valid).

Scalability Further experiments have been carried out to evaluate the possibility of scaling up the proposed localization system to larger buildings with a higher number of rooms. Usually, the localization performance degrades when the distance between the transmitter and the receiver increases. This evaluation relies on the analysis of accuracy as a function of the ratio between the number of rooms in the building and the number of receivers available, referred to as rPr in the following. To this purpose, the localization system has been tested on different datasets featuring an increasing number of rooms per receiver (rPr):

- DB1: 2840 observations acquired in a building with 19 rooms and 7 receivers installed ($rPr = 2.7$);
- DB2: 2130 observations acquired in the same building of DB1 with 5 receivers installed ($rPr = 3.8$);

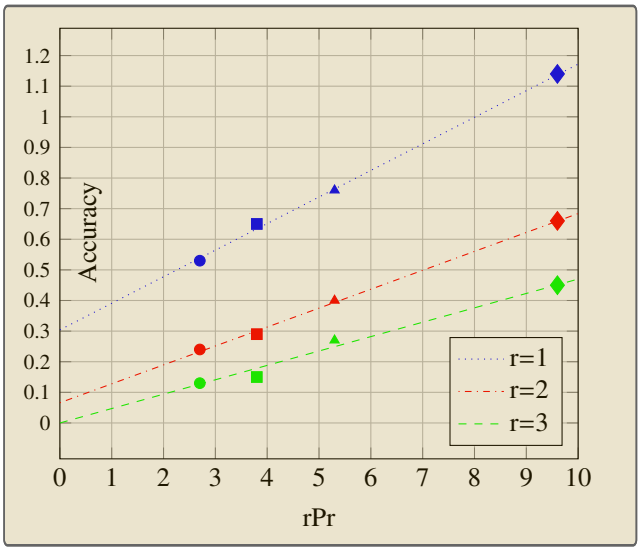


Figure 2.18: The accuracy of the localization system on three different datasets with $w = 1$, $r = 1$, $r = 2$ and $r = 3$ (blue, red and green elements respectively). The accuracy trends as a function of rPr (lines) is also shown. Finally, diamonds highlight the accuracy measured on a dataset with $rPr = 9.6$.

- DB3: the database described in Section 2.0.10.1 with 48 rooms and 9 receivers installed ($rPr = 5.3$).

The results obtained are reported in Figure 2.18, where accuracy obtained with $w = 1$ is reported for each of the three databases and different values of r (the markers in the chart). The chart also reports trend functions (represented as lines) that could help to predict the system behavior at different values of rPr. To validate the observed model, an additional simulation has been carried out by ignoring the signals from four of the nine receivers used in the experiments on DB3 (i.e., A, B, G and I in Figure 2.7), thus increasing the value of rPr from 3.8 to 9.6 (48 rooms with 5 receivers). The accuracy measured with r equal to 1, 2, and 3 is: 1.15, 0.65 and 0.45, respectively. The measured results (diamonds) are extremely close to those predicted by the trend function (1.14, 0.66 and 0.45), thus making us confident that, at least for a reasonable range of rPr values, the trend lines are rather reliable.

Chapter 3

ICT Urban Infrastructures

New ubiquitous and mobile technological urban infrastructures are the key to understanding how the world has been transformed into an entire urban phenomenon [36]. A global networks of signs, values, ideologies and locative media gives us the freedom of spatial mobility and the possibility of creating and recreating places; our daily life is revolutionized by pervasive sensor networks which constantly sense the urban context and allow a hypothetical total control of space.

As an example, we are witnessing an increasing traffic congestion in the urban context. This is mainly due to both the tendency for people to move to the city from neighboring areas and an increase in the average age of the population, as a result of medical, social and cultural advances. It is estimated that more than 50% of the population is concentrated in cities [85]. This phenomenon of congestion along with many other social, political and economic factors makes urgent a rationalization of urban processes to improve the quality of life, energy saving and more generally the sustainability of the planet [111].

Weather sensing represents another key-feature for smart cities. Monitoring weather conditions and environmental features in general allows to better cope with traffic management, event management, crowds management and especially allows to better handle disasters and emergencies due to extreme weather conditions.

The increasing use of sensors in urban environments is making cities smarter. Smart cities are high-performance urban contexts, featuring a constant flow of information between citizens, city stakeholders and city intelligent infrastructures. As explained above, an urban ICT infrastructure can be considered as the brain of the city; it processes sensed information by the body, and coordinates intelligent reactions to deal with the environmental conditions [23]. In order to provide contextualized and location-aware services to each city actor, sensor networks are deployed within the city. The aim of these sensors is to constantly provide raw data gathered from

the city, and consequently enable real-time monitoring, interaction and reaction to the different city events [88, 111].

Along with the spreading of sensor networks, computing and storage virtualization through dedicated ICT platforms is growing as well. Cloud computing relies on the usage of remote hardware and software resources as services. Connecting sensors and actuators to an urban ICT infrastructure enables users to transparently access the sensing networks. This is an Internet of Things (IoT) abstraction [51, 46], in which each sensor becomes an accessible object.

In the previous Chapter a focus on smart city services was posed. In this Chapter the author analyzes the underlying infrastructures. Specifically, the discussion concerns the design of the "City Kernel" [24], a novel solution proposed by the author in order to rationalize the deployment of urban ICT infrastructures and connect them to front-end smart services exposed to different city stakeholders. Then, two novel infrastructures designed by the author are described and discussed. The first concerns urban traffic and mobility while the second addresses problems related to extreme weather conditions in urban contexts [37].

3.1 The City Kernel

ICT infrastructures play a key-role as they are often at the basis of the levers that allow the application of management control to the urban context, in its broadest sense. In particular, sensor networks [51] are the tool that allow to establish a direct and capillary connection between the control and monitoring centres of the city and the entire urban area.

In the future we probably shall need to design infrastructures capable of handling heterogeneous sensor networks in order to integrate the raw data coming from urban area and process them to expose a range of services to the citizen, enterprise, government, and machine to machine [14].

In order to cope with these issues a city should be equipped with some central infrastructure that enables real-time monitoring of each aspect of the urban area (for instance, see the *kernel layer* in Figure 1.1). We refer to such an entity as the *city kernel*, a server-side system developed at the *Smart City Lab*[1], a research laboratory part of DISI (Department of Computer Science and Engineering) - University of Bologna. This system is able able to collect information from multiple sensor networks

[1] http://smartcity.csr.unibo.it

(in the following referred to as subsystems) and to expose multiple services, processing data gathered from one or more of them.

Through a collaboration between Smart City Lab and *Umpi Elettronica srl*[2], a leading company in the field of remote management of outdoor and indoor lighting infrastructures and other smart city technologies, a whole verticalization of the city kernel was developed and tested. Basically, a network of sensors for urban traffic monitoring was linked to the kernel and some useful end-user applications that use the kernel as a content provider were developed.

The core block of such a system should in general expose a list of useful services in the urban context, inferred from some raw data processing [46]. These raw data will be collected from the kernel itself thanks to the continuous communication with many subsystems, each one addressing a specific problem or context. For example, two subsystems may consist of a sensor network for sensing the weather conditions and a sensor network for traffic flow monitoring, while a related service could provide probable points of congestion in the urban network crossing the raw data of the two previous subsystems and subjecting them to some form of classification.

The proposed system exposes a list of services, each one ruled by a public contract that has to be observed in order to call the service itself. The contract usually contains some specification such as expected input data, output formatting and available features description as well.

When a client application calls a remote service functionality the kernel module acts as follows:

- The exposed running service receives the query containing the authentication data and the parameters useful to execute the requested feature.
- An authentication module checks if the client is allowed to call the service.
- On success, the kernel instantiates a *Job Manager*.
- The *Job Manager* instantiates a *Mapping Manager* and a *Data Processor*.
- The *Mapping Manager* retrieves the raw data from the relevant data bases.
- The *Data Processor* processes these data and prepare the response with the appropriate formatting.

[2]http://www.umpi.it

- Finally the *Job Manager* provides the response to the calling client via the service.

Figure 3.1 shows the high-level design of the city kernel.

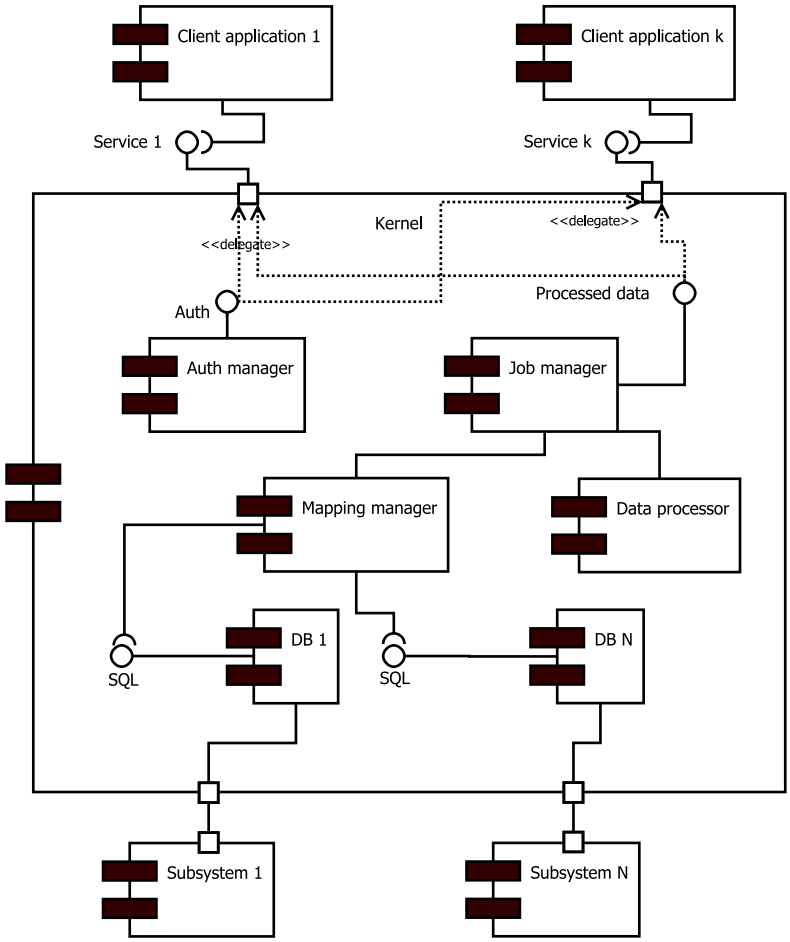


Figure 3.1: This UML component diagram provides a high-level software engineering schema of the city kernel: this module is designed to collect raw data from many different source, here called subsystems, and expose services for the urban context based on the processing of those data.

In the following we show how to expose services based on the raw traffic data gathered by a subsystem consisting of several traffic sensors. In order to reach this goal, the subsystem needs to be linked to the *city kernel* and some procedures able to process raw data correctly need to be designed as well.

The first task is achieved via both Ethernet and PLC connection, while the second one is achieved designing specific methods in the *Job Manager* and in the *Data Processor* components (see Figure 3.1) in order to serve the client application requests.

3.1.1 Requirements of a smart sensing infrastructure

In this section we discuss the requirements of an intelligent information infrastructure that is at the basis of any smart city. The smart city paradigm revolves around integration of existing, conventional infrastructures into a unique intelligent infrastructure that collects data generated throughout the city and provides a unified platform for accessing it [68, 98]. Information and communication technologies and digital networks now widely available in modern urban contexts make this integration possible. The outcome is better management of information, providing each involved stakeholder with local, reliable data access [84, 103]. Such an information system, however, comes with unique challenges.

The amount of information managed, its location-aware nature and the multiple sources spread throughout the city require an innovative use of existing information technologies. In the following we propose the design of such a system. The proposed infrastructure addresses requirements imposed by the urban context on information system design:

- *efficient*: in everyday city life, the information infrastructure will be queried repeatedly by stakeholders and should at the same time collect citizen- and sensor-driven data. So, each task has to be processed efficiently in real-time [103].

- *write-intensive*: the urban context generates a large amount of data derived both from sensors and citizens. These data have to be continuously stored in the information system [3].

- *non-transactional*: data base consistency is not the most important property when dealing with *big data*. Queries can be performed in a loosely-transactional isolation level in order to enhance the performance [101].

- *available*: it is important that the system stays up and running even if overloaded or in case of a failure of one of its components, a likely occurrence in such a complex infrastructure [103].

- *scalable*: the urban context is complex and changes continuously, new sensors could be added and new features could be implemented in the system. It is thus important that the system can scale well [103].

- *fault-tolerant*: the system has to cope with components failure as explained above. No *Single Point of Failure* (SPoF) should be present and the stored data must not be lost in case one of the data base servers should fail [3].

3.2 WTC sensor network for traffic monitoring

In the past few years *intelligent transportation systems* were often adopted by cities governments in order to optimize the traffic flow. The literature shows works focused on algorithms capable of finding traffic patterns in the road grid [69], [73], and articles approaching pervasive and adapting infrastructures able to suggest the best route in real-time, with the aid of capillary distributed devices such as smartphones or tablets [15], or to infer high-level information such as the user direction, needs and habits in the urban grid context [137].

A meaningful issue that must be addressed when designing a system for urban traffic monitoring is the ubiquity of detection points, as the quality and reliability of the designed system is often closely linked to this factor. A widely distributed network of sensors requires an equally extensive infrastructure to grant energy supply to the sensors and to establish a communication channel with them; implementing such an infrastructure involves high costs and relevant political choices that often compromise the feasibility of the entire project. The costs for installing and maintaining the designed network, both for sensors and infrastructure, are of great importance in relation with the municipality decisions and should come with a sustainable urban development, as pointed out in [63].

The traditional inductive-loop sensors need the breakage of the road surface during both installation and maintenance and therefore, in these phases, lead to the hold-up of the urban traffic. They also require dedicated devices (usually deployed to the road boundaries) to allow the communication of the collected data [63]. Similarly, the most recent optical over-roadway sensors or those sensors based on reflection of transmitted energy (like infrared laser radar, ultrasonic sensors, and microwave radar sensors) require accurate installation procedures; in fact, radar sensors need to be properly aligned with lanes and, in general, a separate sensor for each lane is required, although they prevent the breakage of the road surface. Even in this case a receptive device at the roadside is needed, except when the sensor is natively equipped with ethernet or wi-fi interface and the installation area is properly covered [63].

The optical sensor described in this thesis, named WTC (*Wise Traffic Controller*), avoids all of the problems mentioned above. This sensor is in fact installed roadside,

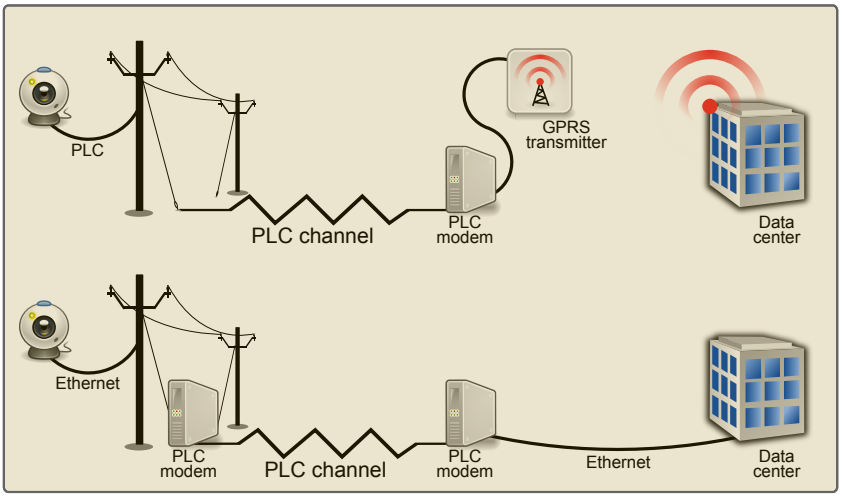


Figure 3.2: A representation of the physical channel used for communication, exemplified by two possible configurations. The main idea comes linking the WTC to the city kernel over the urban power supply.

exploiting the light poles, and therefore does not inhibit the traffic flow nor during installation, nor in case of maintenance. Furthermore, it relies on the electrical network (which certainly reaches the light poles) both for power supply and for communication, thanks to the PLC (*Power Line Communication*) technology.

Without adding any infrastructural element to the urban environment, but taking advantage of the ubiquity that comes with the power supply, this sensor can collect traffic data and periodically communicate them to the remote storage and processing engine (referring to one of the databases managed by the city kernel, as described in Section 3.1).

3.2.1 WTC over PLC

Power line communication was introduced in the 80s to exploit electric power lines as communication channels inside offices and buildings in general, in order to enable communication networks without the need of dedicated physical links installation [126].

The principle that underlies this type of physical channel consists in superimposing a modulated carrier signal on the wiring system. The power distribution system, usually transmitting a low frequency signal (*50-60Hz*) is impressed with a higher frequency signal (*2MHz* to *30MHz*) that adds information to the standard one. An appropriate modem installed to both ends of the channel enable the transmission of the information from the electric channel to a computer, or a data network.

Umpi Elettronica srl has developed a system able to dimmer the lights at the poles and switch off the lights while keeping them powered thus enabling to activate

additional devices at daytime. This achievement comes adding a small device to the power stations. The possibility to deploy various sensors and devices over the outdoor public lighting infrastructure allows a city to be turned into a network capable of collecting massive and relevant information.

The WTC is then directly connected to the power supply, and data are delivered to the first infrastructural node served by ethernet connectivity or by a GPRS (*General Packet Radio Service*) transmitter, which constitutes the last stretch of the communication channel to the remote storage engine, as shown in Figure 3.2.

The server-side responsible for monitoring the WTCs network uses a software module that resolves IP addresses (and also knows the SIM card identifiers eventually installed in the GPRS units). It periodically polls the WTCs to check their state and transfer the collected data to a remote database. After the check is performed, the sensor memory is released, ready to store new urban traffic information.

3.2.2 Wise Traffic Controller

The WTC consists of two main components: a smart camera equipped with embedded software and a communication module located at the basement of the electric pole.

While the first is responsible of traffic data capturing and processing, the second enables the power line communication.

The smart camera is equipped with an embedded system based on CMOS sensor (*Complementary Metal-Oxide Semiconductor*) and a microprocessor for real-time processing of the captured image frames. A volatile memory block, a non-volatile memory block, a magnetometer and an accelerometer are also included, as shown in Figure 3.3. Images are not stored on the device but are just processed in order to extract useful information (i.e. vehicles counting, vehicles classification). This

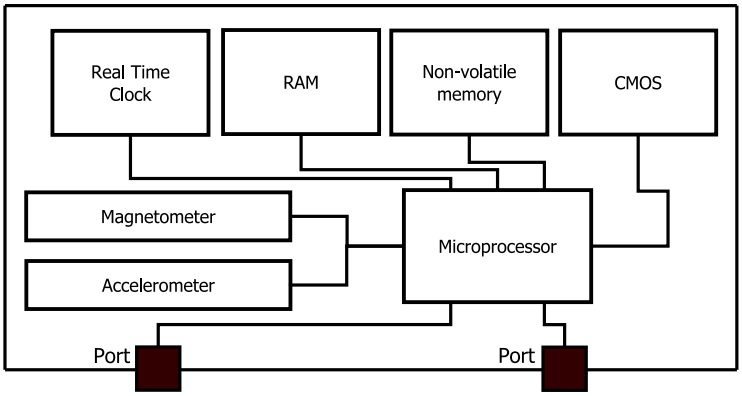


Figure 3.3: WTC hardware components diagram.

is actually a relevant feature: in fact, recording videos or images in the urban context may infringe several existing laws. Legislation about privacy in this context is extremely complex and varies significantly from country to country; therefore, to completely avoid any recording or transmission of videos and images is a great benefit. Furthermore, the absence of transmission of each type of media drastically reduces the bandwidth required to run this network of sensors and enables communication through PLC.

Many techniques have been proposed for vehicle counting and classifying based on optical sensing. In our case, where the camera is fixed and set in a suitable way with respect to the lanes, the most popular methods relate to *thresholding*, *edge detection* [72] and *background frame differencing* [70]. A more extensive list of all the methods designed for this purpose can be found in [58].

The WTC sensor uses a specialized form of background frame differencing optimized to lighten the computational load and satisfy real-time requirements on a processor based on a single *ARM9 200 Mhz* core. The embedded system uses the *virtual-loop* principle, similarly as described in [128]. A virtual-loop is a region-of-interest (actually a small rectangular region) defined in the video image by the technician, covering a traffic lane. Vehicle detection and counting are evaluated just for the pixels included in the small region nearby the virtual-loop, significantly reducing the volume of data to be processed (typically below 5% of the resolution of the image acquired by the WTC sensor).

The software installed on the WTC supports two different operational modalities, respectively for day and night scenarios. The embedded system evaluates the lighting conditions of the acquired image and performs an automatic switch between the two modes in real-time.

The WTC embedded system is capable of:

- counting vehicles travelling in both directions;
- classify vehicles into the classes motorcycle, car, trucks (not in night mode);
- computing the vehicles speed (not in night mode);
- detect traffic congestion;
- simultaneously monitor up to 8 lanes.

Condition	Vehicles	Detections	Error
`Day`	1031	1022	1.07%
`Day, slight shadows`	336	349	3.57%
`Sunset, sharp shadows`	604	672	11.26%
`Night`	253	264	4.35%

Table 3.1: The error rate is computed by comparing the data collected by the sensor with the corresponding video sequences specifically acquired for comparison purposes.

The embedded task of image processing and classification is extremely fast and does not affect the speed of acquisition of the signal by the optical sensor that varies between 25Fps and 35Fps [3].

The behavior of the WTC sensor is for now compromised by extremely bad weather conditions as heavy rain or heavy snow. On the contrary, the well known and hard to solve problem of shadows casted by vehicles on other lanes is addressed by a novel technique called *Vehicle Shadow Detector* (VSD). In the first experiments carried out, VSD allows to remove about 50% of false transits caused by shadows, without removing genuine transits. We are confident that, after optimization, the efficacy of this approach can improve significantly. This algorithm can not be further described as it is under patent pending process [76].

Table 3.1 shows some empirical results based on comparison of data processed by the sensor and some corresponding video sequences recorded during 8 months of experiments in meaningful and different scenarios. Note that the algorithm of shadows detection was not running during these tests.

3.2.3 Smart services over WTC network

For higher efficiencies and cost savings governments will need to focus on a *Service-Oriented Architecture* (SOA) to address the challenge of building a smarter city; on this basis, to provide an integrated platform on which to develop multiple ICT services is reasonably a good model [4].

The connection of the WTC subsystem to the city kernel enables the storage of the raw data gathered in the urban environment and their subsequent processing in order to expose smart-services, i.e. value-added services.

In particular, two services have been designed on the city kernel in relation to the WTC subsystem. The first represents a city-to-citizen smart application designed for

[3] Frames per second.

mobile devices; the second is a city-to-enterprise tool instead, designed to support the government or the company responsible for the urban traffic network.

According to future needs, the government could develop other services, always accessing the same data processed in the kernel, and possibly integrating them with information provided by other subsystems in order to reach even more powerful and pervasive applications.

3.2.3.1 Mobile traffic control

This service allows citizens to avoid queues and traffic jams during a trip on their own transport, choosing the more flowing path. This is achieved through the processing of the data concerning the real-time traffic situation inferred from the WTC subsystem, which allows to identify critical points and unusual traffic situations.

The application, designed to act as an interface with the citizen, has been developed for Android OS; in fact, today the Google mobile OS holds a significantly large smartphones market share [122]. As launched, the application instantiates two threads: one to handle the communication with the remote service exposed by the kernel and one for the calculation of the current geographic coordinates (thanks to GPS and wi-fi sensors). After a successful georeferencing, a complete list of the surrounding WTC sensors is presented, sorted by the distance from the user. For simplicity, all these data are also delivered through an interactive user-centered map. Each sensor shows a set of possible alarms (jams, queues, anomalies) together with some parameters about the standard traffic flow (such as last hour vehicles counting, vehicles average speed and so on).

A schematic representation of the application and the underlying infrastructure is shown in Figure 3.4.

Such an application also allows to compute optimal routes over the road network avoiding inconvenient traffic nodes, in an automatic and transparent way.

When the user types the destination he wants to reach, any point on the map where a WTC sensor has revealed anomalies is detected. That way, points to be avoided during the processing of directions between the starting point and the destination can be identified. The tools for calculating the routes on the map are based on Google Directions Api [40].

3.2.3.2 Visual WTC

This service, developed by Umpi R&D, provides a web interface and is designed to be used on a personal workstation on a big screen (e.g. in a control center).

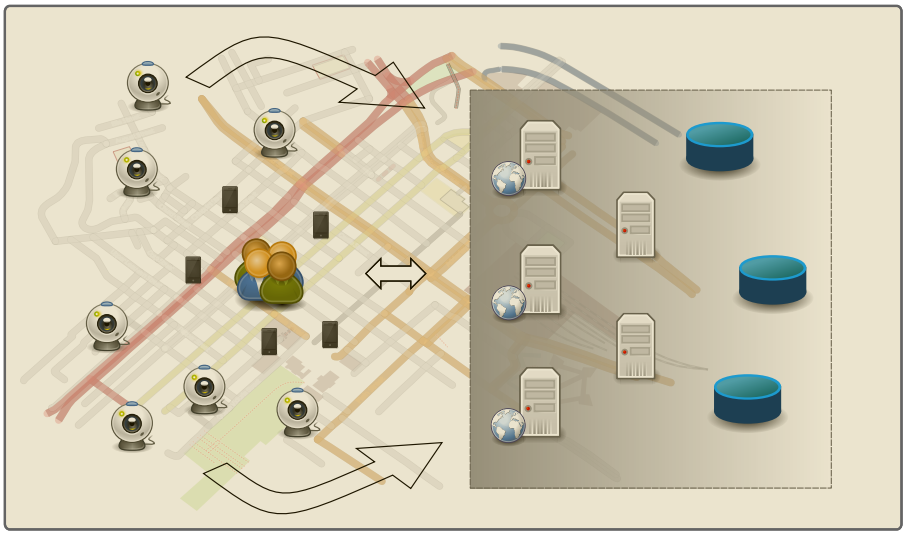


Figure 3.4: The WTC sensors scattered in the urban context constantly communicate the raw data about the urban traffic to the city kernel storage servers. The kernel's application servers expose a service that allows Android mobile clients to display real-time traffic information to the citizens.

Here all the WTC sensors related to the docked subsystem are shown on a map. Each sensor is supplied with the latest data about traffic performance (similarly as explained in 3.2.3.1). This software is also equipped with advanced analytical processing functions that allow to extrapolate different statistics about the traffic flow. It is possible to perform studies on the average traffic load on certain routes, to obtain the rate of congestion of certain nodal points, on a daily, weekly, monthly or arbitrary basis. Similarly, it is also possible to discern the statistics according to working days or holidays and so on.

Specifically, for each WTC, the software keeps track of:

- Omni-directional traffic for all types of vehicles.
- Omni-directional traffic for each single type of vehicle.
- Traffic for each single direction for all types of vehicles.
- Traffic for each direction and each type of vehicle.
- Traffic for each virtual-loop for all types of vehicles.
- Traffic for each virtual-loop for a single type of vehicle.
- The average cruising speed.
- The average cruising speed for each single direction.

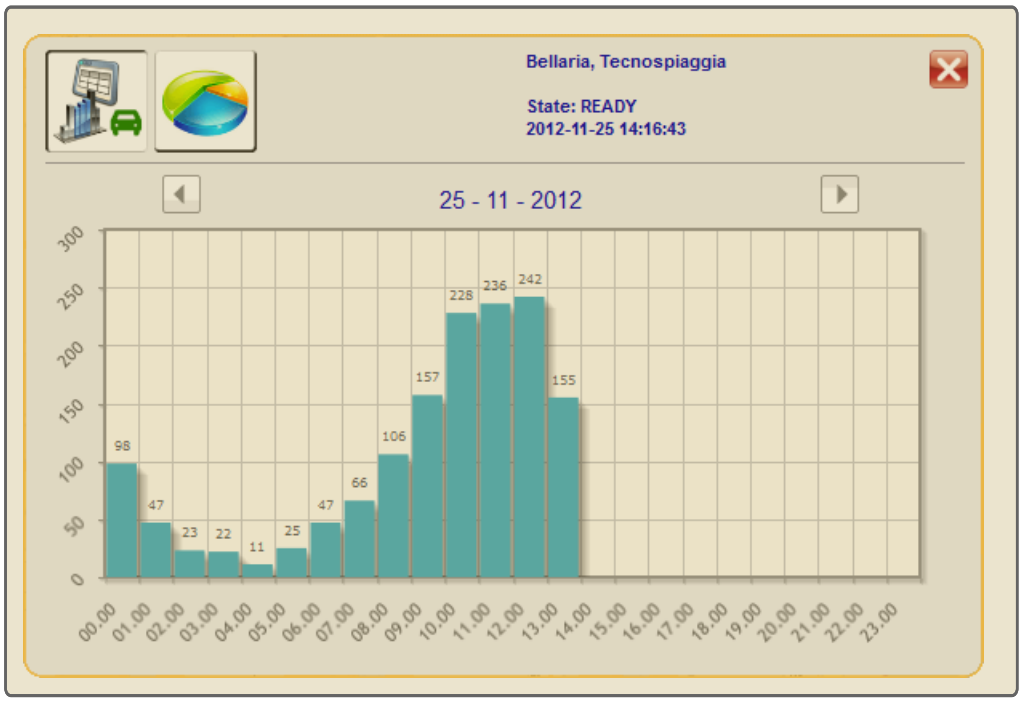


Figure 3.5: A screenshot of the Visual WTC web-based service in action.

- The average cruising speed for each single virtual-loop.

Note that it is possible to set the thresholds in order to adjust the trigger alarms related to traffic congestion.

This tool allows the government or the company that is supposed to control the traffic flow to better design the urban road infrastructure, the road signage and the traffic management policies in general, thanks to the provided analyzes.

Thanks to the versatility of the underlying infrastructure, it is possible to run limited tests involving a single urban area moving WTC sensors from a light pole to another, in order to keep the costs as low as possible.

A screenshot of this application is shown in Figure 3.5.

3.3 Extreme weather sensing for safer cities

As in many other smart city scenarios, water management can greatly benefit from the use of smart software platforms and widespread sensing, both during regular operation and scheduled maintenance, and in emergency/unforeseen situations. In particular, insights from the intelligent combination of disparate data received from various sources (from sensors and metering devices to crowd-sensed data) can significantly help to better manage the water infrastructure, assets and operations. In this context, it is crucial to design systems that allow the analysis of heterogeneous

sources and the integration of social data, with a particular focus on a distributed sensors and information infrastructure. Important parts of the system are advanced visualization and correlation tools.

Urban water management becomes crucial during heavy rain events when flooding may occur, causing damage and social disruption [6, 118, 120, 133] as a result of short-lasted, intense precipitation peaks. Since intense rains are expected to become heavier and more frequent as a result of climate change, pluvial related damage is expected to increase in the future [6, 10, 125]. Urban water systems respond rapidly to rainfall as a result of their high degree of imperviousness. For this reason, localized, real-time information is required to react adequately during emergency situations. A system that allows quick, coordinated, and to the point intervention during extreme rainfalls, based on the aggregated information and with a high level of automation, is needed. Intelligent use of information from multiple urban sensors can provide detailed, real-time insights into the urban flooding process and allow citizen's and local administration to take timely action and prevent urban flooding impacts. The design of such a system, based on the existing sensor infrastructure of the city of Rotterdam (The Netherlands) is proposed contribution within this thesis. While the technical tools used in the design are not innovative *per se*, the challenges imposed by the urban environment are unique, and an integrated approach is adopted in the proposed solution.

Many solutions have been recently proposed for the design of ICT infrastructures for smart cities. Such proposals usually aim to define a city brain [24] able to collect raw data from all around the urban context, and expose value-added services to citizens and other stakeholders. These models are designed with integration of both human and sensor-based information in mind [81]. Human movement dynamics in crowded cities have been discussed as well [77]. Cell phone traces represents a widespread source of information to understand urban mobility patterns, and can be useful for risk and disaster management and prevention [19]. Spatio-temporal dependence between aggregations of mobile network traffic and distributed weather data have been also previously explored statistically, allowing for a context-aware evaluation of the link between environmental and social dynamics [106]. Cellular signals were studied for modeling rainfall distributions at a national level. The signal interference caused by rain drops was measured against the expected signal strength. In this way, the microwave antennas, initially intended for cellular communication, were used as a network of weather sensors delivering spatio-temporal distributed data [94]. Innovative, real-time, distributed, and cost-effective weather sensing in poorly

instrumented areas is addressed in the Trans African Hydro-Meteorological Observatory (TAHMO). The joint use of tailor-made sensors, RaspberryPis, and IBM's IOW platform have been tested as part of the TAHMO activities. The resulting gadgets proved to be able to deliver real-time weather data via the internet[4]. A citizen-based observatory of water systems, which allows citizens and communities to become active stakeholders in information capturing, evaluation and communication, is being developed within the project WeSenseIt[5]. At Delft University of Technology, the SHINE project aims to enable gathering, processing and interpretation of multiple data sources for answering information needs. One of the involved projects focuses on next generation ground-based meteorological radars for highly detailed urban rainfall measurements [121]. In the RAINGAIN project various radar configurations and hydrodynamic softwares are implemented and tested for high resolution urban rainfall estimation and flood prediction to support urban pluvial control[6].

3.3.1 An integrated infrastructure for rainfall sensing in Rotterdam

In the following we describe information sources relating to a heavy rain event in Rotterdam on October 12-14, 2013. Rotterdam weather monitoring infrastructure is composed of a number of rain gauges installed at different locations in the city, as well as a weather radar network. This sensing network is currently scarcely integrated and logged data are not easily accessible during an emergency. Therefore, in this thesis, the author proposes a reliable, efficient and low-cost ICT infrastructure that takes information from all relevant sources, including sensors as well as social and user contributed information and integrates them into a unique, cloud-based interface [37]. The proposed infrastructure will improve efficiency in emergency responses to extreme weather events and, ultimately, guarantee more safety to the urban population.

3.3.1.1 Sensing extreme rainfall

The city of Rotterdam was hit by a heavy rain event during October 12 to 14 2013, with a total rainfall volume of approximately 130 mm. The annual precipitation average in this city is around 800 mm; a sixth of the yearly rains fell within just 3 days. Local flooding in various the city caused damage, social annoyance and interruption of traffic and economic activities at certain areas of the city.

[4]http://tahmo.info
[5]http://citi-sense.nilu.no/Project.aspx
[6]http://www.raingain.eu

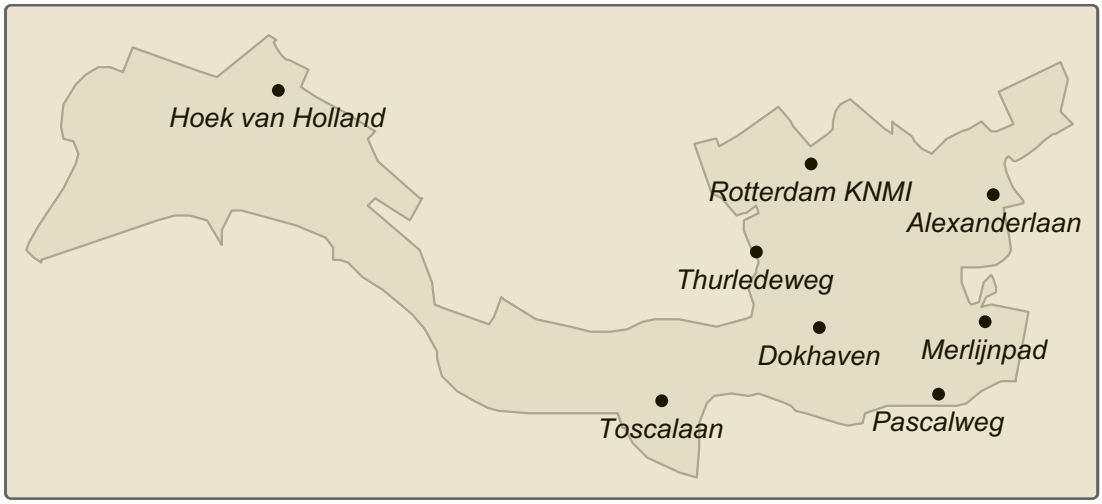


Figure 3.6: Location of Rotterdam disdrometers and the *Rotterdam KNMI* raingauge.

Located in the Delta of the New Meuse river, Rotterdam is highly vulnerable to extreme weather conditions. Rainfall is expected to become heavier as a result of climate changes especially in the coastal region. The 600 thousand inhabitants city is the biggest port of Europe and accounts for a considerable fraction of the Dutch economy.

A set of the available information describing such extreme rainfall events is presented in this section. The accessibility and usefulness of this collection provides a picture of the challenges to be addressed by the distributed information infrastructure here proposed. Rotterdam is instrumented with a set of rain gauges, and is covered by the weather radars of the Dutch National Meteorological Institute (KNMI). Besides, the Municipality and the local Fire Brigade provide platforms for receiving citizen complaints that are used for responses during the occurrence of extreme rainfall events. Additionally, the urban topography has been modeled into a a highly detailed digital elevation model (DEM) that allows to deduct slope aspects and surface drainage directions. The related information is described and visualized for the example of the heavy rain event of 12-14 October, 2013.

Rain gauge measurements Rotterdam is covered by various weather sensing platforms that record rainfall measurements. KNMI provides near real time (delay of approximately 6 minutes) rainfall measurements from an automatic electrical rain gauge located at Rotterdam airport, at a temporal resolution of 10 minutes. This rain gauge is part of a national network that includes 32 automatic weather stations. The network collects rainfall data of reliable quality; studies have reported measurements with up to a 5% underestimation when compared to manual rain gauges [17].

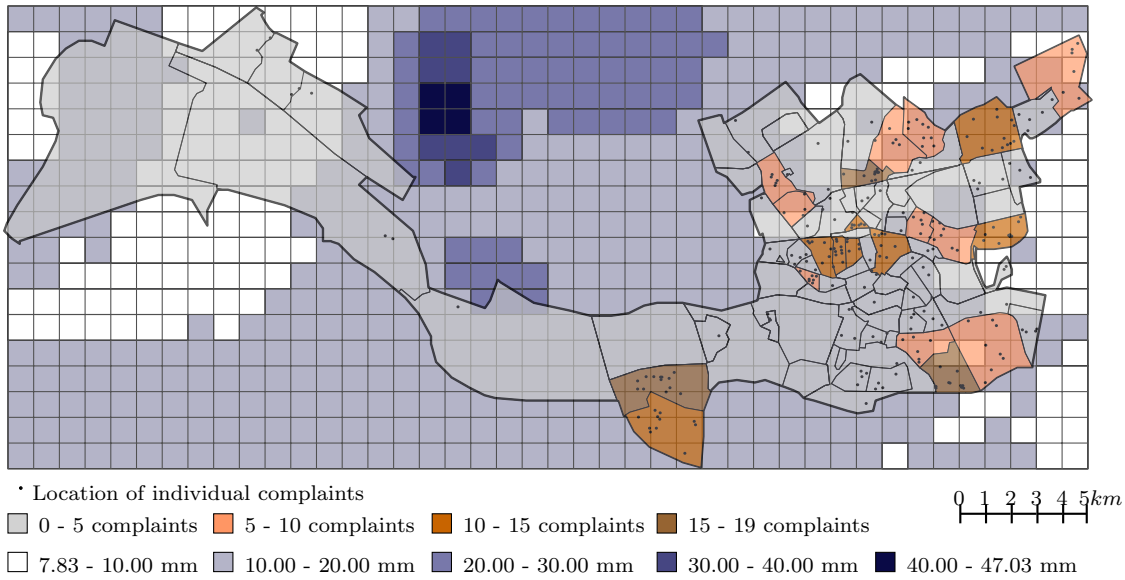


Figure 3.7: View of Rotterdam districts between October 12 to 14 2013. Black dots indicate the location of citizens reporting rain-related incidents. Highest rain intensity at each pixel is represented with different blue levels. Sums of complaints per district are illustrated with different red levels.

Data are available under a licensed access through a ftp connection.

Additionally, the study area is covered by a network of 7 Thies laser disdrometers. These rain gauges yield 1 minute time-step measurements of rainfall volumes, with levels ranging from 0.005 to 250 mm/h. It uses a laser sensor to detect signal interferences due to particles falling through a laser beam. Measurements are automatically sent to a central database. A recent study compares disdrometer measurements with rain-gauges of the KNMI national network. Results showed that monthly accumulated rainfall of 4 of the laser disdrometers are highly correlated with the KNMI rain gauges, while two of the laser disdrometer presented correlations with the KNMI rain gauges as low as 0.5 and 0.2 [55, 92].

Currently, an additional network of 10 automatic weather stations located in Rotterdam is being updated. This network will be able to provide rainfall measurements with a temporal resolution of 5 minutes. The location of the stations is chosen so as to prevent effects of rain shadows and turbulences due to the layout of buildings and other urban characteristics.

Figure 3.6 presents the locations of the seven laser disdrometers around Rotterdam and the KNMI rain gauge at the Rotterdam airport.

Weather radar KNMI provides a weather radar product that is processed from two C-band Doppler radars [113]. This imagery product has a spatial resolution of 1 Km^2 and a temporal resolution of 5 minutes. Figure 3.7 shows the grid which KNMI weather radar imagery pixels align to. Raw data is provided in near-real-time at a radiometric resolution of 8 bits. Digital values represent a logarithmic scale of radar decibels [1]. Under particular meteorological characteristics and at distant ranges the weather radar tends to underestimate precipitations [93]. Besides, radar measurements may not reflect the weather conditions at ground level [93, 102]. The advantage of radar-derived rainfall data as compared to point measurements by rain gauges is their spatial coverage at areas poorly covered with weather stations [113]. The Rotterdam region will soon be covered by the range of an X-band weather radar which aims at better depicting the variations of rainfall at urban spatial scales.

Digital elevation and urban watersheds models A DEM is available for the city [110]. The DEM was produced using Light Detection and Ranging (LiDAR) of ground levels from an aerial platform. The DEM delivers a spatial resolution of 0.5 m $\times$ 0.5 m, a vertical precision of 5 cm, a systematic error of 5 cm, a random error of 5 cm, and a minimum precision under two standard deviations of 15 cm [127, 110]. This DEM has been used for delineating overland flow paths. Such paths trace the directions that water tends to follow after rainfall due to urban slopes. Figure 3.8 shows an example of the representation of the elevations and flow paths in a street of Rotterdam. The paths follow the slope until they reach a urban water body (e.g. a canal or a lake).

Human sensing The Municipality of Rotterdam maintains a platform for receiving and registering citizen calls about incidents in the city. Registering those calls consists of setting a unique identifier number, transcribing the complaint information into short textual descriptions and performing a first-stage classification of the corresponding incident categories, stamping the time, and annotating the reporting address and the name of the caller. The complaints classified as 'water- and sewer-related', were made available by the Municipality for academic and research purposes. Due to privacy issues, these registers are not publicly available. The city's Fire Brigade service maintains a similar platform. Calls from citizens reporting diverse types of incidents are registered using a unique identifier number, the address of the incident, an initial classification of the complaint, a brief description of the incident, and a status indicating whether the issue has been attended to and solved. The registry is

Figure 3.8: 3D representation of DEM and overland flow paths in Rotterdam.

available in a website that provides geographic visualizations of individual reports. This service does not provide machine-to-machine interfaces to access the registries [2].

During the heavy rain event of October 12 to 14 2013, more than 600 citizens telephonic complaints referring to rain-related incidents were registered in Rotterdam. Around 300 calls were received by the Municipality and another 300 by the regional Fire Brigade. Both data sets were made available for research. Figure 3.9 presents a time series of the hourly complaints recorded by the Municipality and the regional Fire Brigade, compared to the hourly rainfall volumes measured by the KNMI rain gauge at the Rotterdam Airport. The figure shows that, while most of the rain fell during October 12 and 13, most of complaints occurred during October 13 and 14. Total rain volume measured by the rain gauge at the airport during October the 13th was 41.85 mm; on the 12th it was 50% of that, 6% on the 14th. Municipal complaints, on the other hand, rise from 15 on the 12th, to 171 and 139 on the 13th and 14th respectively. The peak of complaints made to the Fire Brigade occurred on the 13th, with 268 reports, up from 6 on the 12th, and down to 22 on the 14th.

3.3.1.2 Information system architecture

Weaknesses in spatial coverage, accuracy and timely availability of environmental and water-related measurements can be overcome if a multiple data sources are combined into integrated information products. Near-real time rainfall from different sensors could be cross-checked 'on-the-fly'; high rainfall intensities can be pin-pointed to areas reported in citizen's emergency calls. Impacts due to heavy rain can be better managed if a distributed, real-time accessible information infrastructure is available for citizens and authorities. Citizens are familiar with complaining and emergency

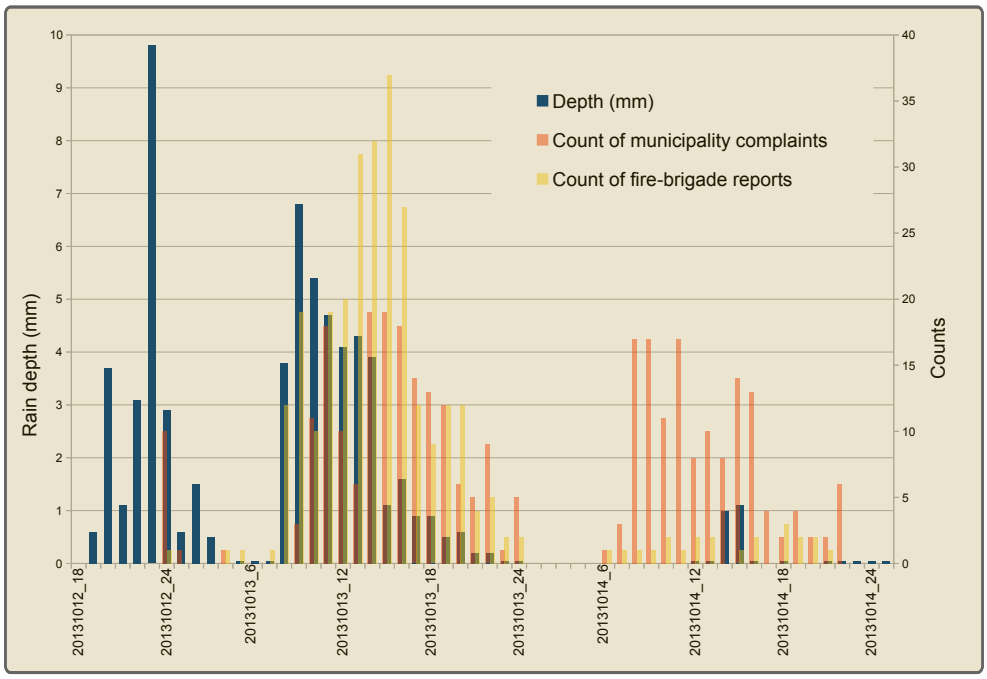


Figure 3.9: A comparison of hourly rainfall measurements with Municipality and Fire Brigade reports.

telephonic lines. Social networks, media in the Internet, and smartphone-gathered data can be also considered as sources to be analyzed. For this reason, building an information system on this basis might yield a valid tool for people to react during extreme rain events. Complaints may be missed when call centers get overloaded by calls, particularly during the peak of extreme events. If there are human errors in the interpretation of citizen reports, complaints registries can be wrongly related to rainfall when it cause might have been other. Besides, receiving citizens' reports about rain-related incidents currently depends on the presence of officers picking up the phone at the call center. If there is nobody on shift, the calls are missed. An internet-based application, accessible via desktop and mobile devices, could manage the register of citizen reports even under multiple requests for service during a heavy rainfall peak, or at night. While the city of Rotterdam is covered by an extensive infrastructure of weather and water-related sensors, it currently lacks an integrated system for managing information generated by these same sensors.

Such a system helps both citizens and institutions to better cope with pluvial-related events, and provides a quick interface for reporting the problems the city is facing. Aggregating data from different sources means that each complaint or event can be visualized on a map on the fly, allowing citizens to help each other, to know locations of pluvial floods and allowing Fire Brigade and Municipality to hold a high level view of the overall situation. For instance, Figure 3.9 shows that for the 12 to

14 October event, complaints were mostly grouped a few hours after the rain fell. A constant overview of the rainfall trends could help Municipality and Fire Brigade to schedule a staff-up in order to cope with upcoming calls. In order to implement a real *living lab* [35, 112] to further collect data, citizens could be involved by means of social network feeds. For instance, the Municipality could define and promote a common *hashtag* (on its Twitter profile) for rain events, and the system should collect and process each tweet containing that hashtag in order to provide stakeholders with location-based, punctual, citizen-sensed information.

In order to collect and visualize citizen-based information, application servers of the back-end infrastructure have to crawl specific social network feeds. Call centers have to provide a simple *web service* to expose complaints content with a machine-to-machine interface. Retrieving and interpreting the right information from social media is a non-trivial task. Not everybody will use the relevant hashtag, and information contained in each feed should be also interpreted automatically in order to combine it with that from sensors. These problems are beyond the scope of this work; here we focus on the overall system infrastructure and on integration of already deployed sensors. In order to integrate rain gauges and allow local management of information before data are collected by the information system, we need to enhance computation and communication capability of each sensor. In the following an installation of autonomous units at each rain gauge is proposed: this allows to collect locally data generated by each rain gauge, and format it according to the standard imposed by the information system interface before sending it over to the system back end. Having computation units at the sensors realizes the sensor network paradigm, in which sensors are able to communicate and interact with each other.

Achieving local computation with Raspberry Pi In order to connect the rain gauges directly to the information system, we propose installing on the spot low-cost, low-power computational units such as single-board computers. Installing one directly where each rain gauge is allows us not only to achieve a better information management of the local data generated by the sensor, but also to build a network of independent units able to react to malfunctions on a single node. Moreover, such computational units are not restricted to the rain gauges, but can be installed on any other sensor that will be integrated into the system in the future. A computational unit is in fact independent of the kind of sensor it is attached to, and provides therefore a standard platform and a uniformed access to information through a heterogeneous network of sensors, which may therefore be integrated seamlessly into the system.

The computational unit may also act as an aggregator of sensors: once such a unit is attached to a rain gauge, for instance, other sensors and devices can be installed in place easily, without requiring further integration, which would be needed instead if each sensor had to rely on its own proprietary communication system.

We propose to implement this system using the popular Raspberry Pi board. A Raspberry Pi is credit-card-sized single-board computer designed to be particularly power-efficient and low-cost. A typical board consumes only 1.67W under stress (for instance, when recording videos) and less than 1.07W when idle.[7] It mounts a Broadcom BCM2835 system-on-chip (SoC), composed of an ARM1176JZF-S 700 MHz processor, a VideoCore IV GPU, and 512 MB of RAM. The Raspberry Pi provides connectivity through an Ethernet port, but for the purposes of this work, we propose to use the urban Wi-fi network, which covers broad areas of the city and provides reliable, city-managed Internet access to the city ICT infrastructure. Additionally, Raspberry Pi includes eight general purpose I/O (GPIO) pins at 3:3 V, from 2 mA to 16 mA, which can be controlled with tailor made software. These on-board GPIOs allow for lower-level interfacing to electronics such as the ones in weather sensors. Taking advantage of the flexibility of the chosen platform, the setup can be completed by installing a remotely-activated camera sensor into the Raspberry Pi. This provides control centers with a remote view of the rain gauges and, since they are located in strategic points of interest, complements the city surveillance camera infrastructure. In order to face unexpected power outages (frequently occurring during extreme weather events) we also consider installing a back-up power unit that guarantees continued operation even during a blackout[8]. In Table 3.2 we summarized the costs of a single unit, including optional components.

Micro Station		
Component	Optional	Market Price ($)
Board	No	35.00
Battery	No	49.50
Wifi dongle	No/Yes	25-35.00
Camera	Yes	24.99
Total cost		85-155.00

Table 3.2: Basic and optional components for each micro station.

[7]Data from `http://raspi.tv/`

[8]It is useful to note here that rain gauges are power-sufficient. It would be important however, even if it goes beyond the purposes of this work, to evaluate network accessibility during a power outage, as the network infrastructure may be power-dependent.

Each single board can transmit data gathered by a local rain gauge (or, in the future, sensors of other nature) to the information system back-end (described in the following), but may also communicate directly to other boards. A scenario in which the boards pre-compute aggregated data locally based on their respective positions realizes a location-aware sensor network. While the rain gauges available in Rotterdam are generally high-precision, a future research could investigate the deployment of a higher number of lower-precision, inexpensive sensing stations, also based on Raspberry Pi, such as the ones used by the Trans-African Hydro-Meteorological Observatory (Section 3.3). The ICT infrastructure here proposed allows seamless integration of new units, enabling future realization of an heterogeneous and distributed Environmental Sensor Network (ESN) [43]. In any future deployment of new sensing nodes, an efficient geographic distribution may reduce significantly the number of units needed to achieve the same aggregated precision by means of local unit-to-unit communication [66]. Finally, it is notable that installing full-fledged computational units may introduce the risk of external attacks. We believe, however, that the added benefits of the platform largely overcome this risk, which may reduced through an adequate firewall and security policy aimed at restricting access only to authorized addresses (for instance, using white-lists of IP's).

Information system back-end As discussed above, an information system suitable for smart cities scenarios should rely on a distributed, highly-available and reliable infrastructure. Here, an architecture for such a system that is suitable for weather information sensing in a large urban area is proposed. The scenario we are approaching hold the following features: it should be up and running 24h/7 as sensors and other data sources provide the back end with data continuously. For the same reason, it can be referred to as a *write-intensive* application. When a heavy rain event occurs, sensor inputs are joined by an intensive rate of human sensing, like the ones described previously. Moreover, during such an event, both citizens and institutions tend to use intensely each service provided by the infrastructure. All of these factors contribute to produce a peak in the workload of the whole infrastructure. Moreover, as this infrastructure is intended to be up and running especially during heavy rain events, each exposed component should cope with rain-driven failures. In order to obtain a fault-tolerant system we plan not to have any SPoF within our architecture. This goal is reached by providing *loosely coupling* between nodes of the system. Each request to the back-end is handled by several pools of interchangeable nodes. As shown in Figure 3.10, each request goes through a load balancer and is

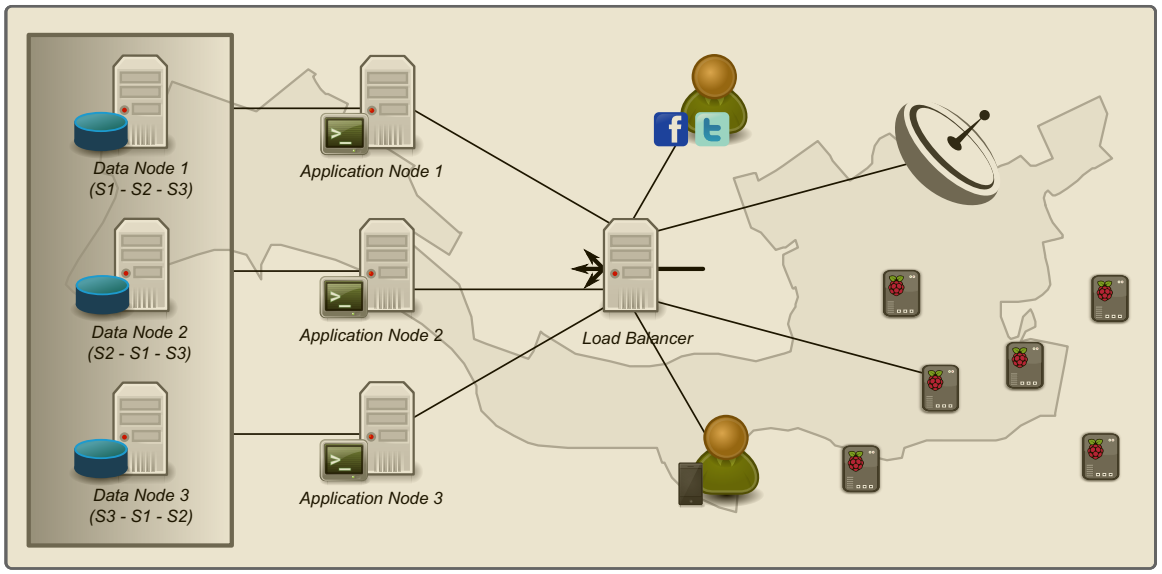


Figure 3.10: Information system architecture.

then passed to an application server, i.e. a node that performs computational tasks like pre-processing of received data. The request ends its route at a data node, a data base server machine where the data are stored. The storage policy that enables data nodes interchangeability and requests concurrency is discussed later on.

The basic idea is that, whether a node would fail, the requests addressed to it can be served by another node belonging to the same pool, without any system failure. The availability goal is thus accomplished. In order to cope with peaks of requests, the system is designed with load balancing in mind. As all the back-end components can be deployed within the city itself, we do not have to deal with latency problems. Basically, for the load balancer, it is not important to address each request to the nearest application server. What is important is that the number of requests each application server handles is balanced among each pool's nodes. This configuration can be reached via a *DNS Round Robin* load balancing approach. This technique consists in configuring the *Domain Name System* (DNS) so that requests are addressed not to a single IP address, but to one IP address among a list of addresses that point to servers hosting identical services. The list is permuted for subsequent lookups. Thus, the load balancer shown in Figure 3.10 should not be intended as a possible SPoF for the system, as it relies on DNS, that is natively distributed and fault-tolerant.

In a distributed write-intensive scenario, a common design pattern for data storage is *Sharding* (from shared-nothing). This technique consists in dividing the data into several independent blocks, and assign each block to a separate data base server. That way, requests from application servers can be addressed in parallel to each per-

tinent data node. This model needs a backup policy to prevent loss of data when a node crashes. The proposed application is not transaction-intensive: data base consistency, crucial in traditional information systems like banking ones, is not a priority when dealing with non critical raw information, as often happens in Big Data scenarios. In those contexts, traditional ACID-oriented DBMS (Atomicity, Consistency, Isolation, Durability) are replaced with BASE-oriented DBMS (Basically Available, Soft-state/Scalable, Eventually Consistent). Thus, when dealing with weather information, queries can be performed in a loosely-transactional isolation level (giving priority to availability and scalability) in order to enhance the performance. Given this, a hybrid sharding solution can fit our needs, sometimes referred to as *active backup nodes*. Instead of storing a sole block of data into each data node, the whole dataset is stored, still divided in blocks. For each block, only one node will act as master, and the others act as slaves. So, when all data nodes are up and running, they act as in a classic sharding environment. When a failure occurs, one of the remaining data nodes becomes master for the block managed by the lost node. Thus, the data node pool is able to continue operating as long as at least one node is alive. With such a solution, both performance boosting (efficiency), fault-tolerance and data recovery are achieved. Note that in order to achieve good performance in a write-intensive scenario, data propagation between each data node is intended to be asynchronous. This is why hybrid sharding solution would not perform well in a transaction-intensive environment, where dataset's consistency holds a higher priority.

By design, we can scale up such a system as needed adding application and data nodes as required, thus achieving the scalability requirement. Two scenario are feasible for the application servers. They could be deployed as pay-per-use servers in the same cloud Infrastructure as a Service (IaaS) platform used for data nodes or they can be hosted directly within each stakeholder headquarter building. In that case, each building should be equipped with a dedicated connection to the IaaS provider hosting the data nodes.

3.3.1.3 System evaluation

The proposed Information System design provides a feasible and reliable way to integrate available, heterogeneous data sources, and therefore allows smarter detection and management of urban flooding. The design relies on existing technologies, and focuses on an affordable and flexible integration of both sensors and crowd-sensed real-time data. Though some of these data have real-time availability, they are dispersed. This limits their use for integrated analytics such as metrics and visualizations

describing the likelihood, timing, and location of urban flooding impact clusters. The presented architecture offers the opportunity to both integrate available information sources and provide reliable access in real time. Thus, integrated and automatic analytics and visualizations become available to support urban flooding management. An interesting analytic to consider is the trend in complaint reports for a street in a critical urban subwatershed at a certain rain intensity. This metric can be used to fire an alarm when a threshold is passed. This kind of threshold can be derived from the historic impacts associated with topographic characteristics, rainfall and complaints levels. The alarm can be used to directly deploy additional staff for coping with problems at a definite position and time, avoiding further inconveniences, and intelligently allocating emergency response resources.

Additional crowd-sensed information can be easily integrated into the system. Twitter posts can be crawled on the fly on the basis of the location and textual content of reports from the most complaining streets or highest intensity rainfall districts during an extreme event. This approach overcomes the limited spatial resolution that Twitter posts have, enabling their use for smart emergency response. In this way, this type of social network data, which benefits from reports made by citizens moving through the city, can be connected to the smart sensing platform. Instantly and publicly available visualization of complaints and tweets relating to urban flooding impacts can enhance a more lively interaction of citizens on the emergency response to heavy rain. To take further advantage of Twitter during heavy rain events, the architecture can automatically post tweets from a dedicated account, or from the account of the head-officer responsible for urban drainage at the municipality or the waterboard. Those posts can not only include links to analytics and visualizations to allow citizens to collaborate on reducing urban flooding impacts, but also a hashtag to invite citizens to tweet additional information such as pictures, qualitative descriptions, and locations of rainfall-related problems. As well as for Twitter, the proposed architecture is able to retrieve information from a smartphone application to complement the already established call centers for citizens' complaints of the municipality and the fire brigade, standardizing and enhancing the spatio-temporal resolution of descriptions of urban flooding impacts.

Power management in installed sensors, and sensing reliability on sensors failure, can be smartly improved by the proposed system. The computational capability of Raspberry Pi opens the way to a number of enhancing strategies for the system's reliability. It is already mentioned that Pi enables several sensors to be plugged into a single computational unit. Let us consider a network of Raspberry Pis, each one

Power saving evaluation			
System	Sensors considering 20% communication	Time on, % (rain percentile)	Yearly power consumption
Old	Rain gauges (0.83 W)	100%	727.08 kWh
New	Rain gauges (0.83 W) Raspberry Pi (1.07 W)	8% (0th)	133.16 kWh
New	Rain gauges (0.83 W) Raspberry Pi (1.07 W)	4% (50th)	66.58 kWh
New	Rain gauges (0.83 W) Raspberry Pi (1.07 W)	2% (75th)	33.29 kWh

Table 3.3: Power consumption evaluation according to rainfall intensity percentiles from Figure 3.11.

equipped with a rain gauge, a humidity sensor and a smart camera. If a failure on the rain gauges should occur, the Raspberry Pi could start processing the smart camera signal with a pattern recognition routine able to detect rain conditions with respect to a predefined background image. That way, the server would continue to receive rain information transparently, despite of the rain gauge failure. The humidity sensor could be queried in turn after a camera failure. Moreover, as these computational units are able to communicate and interact intelligently with each other, and not only to the back end as a regular sensor would, they can be programmed to react to the failure of a whole unit. For instance, after one or more Raspberry Pi installed in a single area failed, the system could be informed by the others of the failures and can carry out several actions: a more frequent sampling could be required from the radar for that area, and Municipality and Fire Brigades headquarters could be alerted to serve the calls from that area first. This example clearly illustrates how combining location-aware sensors data and human sensing together can enhance rescue strategies during extreme rainfall and other related events.

In the following an evaluation of some of the benefits of the proposed system is proposed. It focuses on reduced power consumption, faster availability of information and increased spatial resolution. The power consumption can be reduced by keeping most rain gauges off when no rain above a threshold is sensed by a small set of active rain gauges and the rainfall radar. Using the historic rainfall intensity occurrence

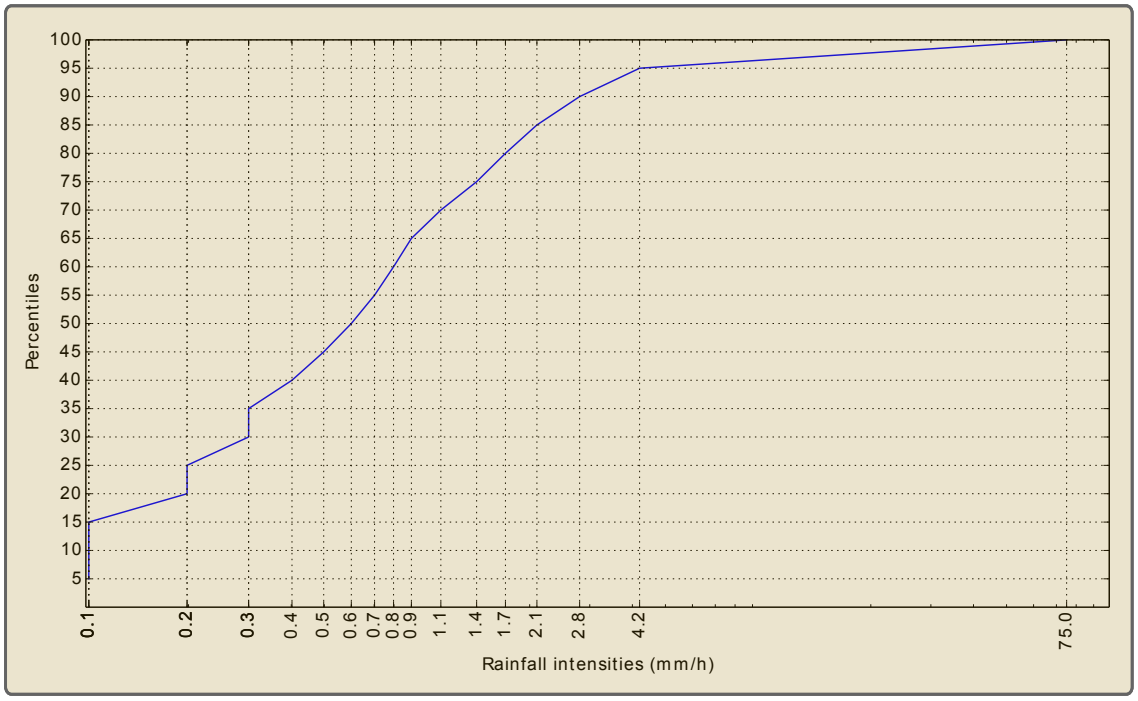
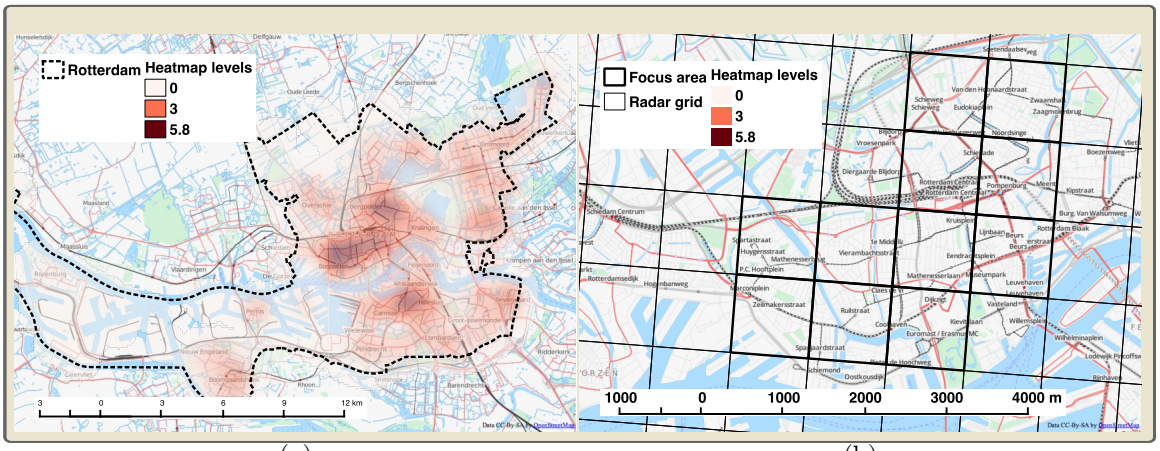


Figure 3.11: Percentiles of rainfall intensities in Rotterdam. A series of rain gauge measurements, made in Rotterdam between April 2003 and March 2011, is used to calculate the percentiles. The series has a 10 minutes temporal resolution. From the total of 420481 timesteps during the eight years analyzed, 34156 recorded rainfall intensities different than 0mm/h. That is, close to 8% of total timesteps registered rainfall. Within such wet timesteps, 50% registered intensities equal or higher than 0.6 mm/h, and 25% equal or higher than 1.4 mm/h. This means that in terms of the total dry and wet 10 min timesteps, $8\% \times 0.50$ registered precipitations equal or higher than 0.6 mm/h (50th percentile), and $8\% \times 0.25$ did it equally or above than 1.4mm/h (75th percentile).

percentiles 0th, 50th and 75th (see Figure 3.11), each RaspberryPi-based sensing station can achieve the power savings listed in Table 3.3. A Raspberry Pi also allows to compare different sensors' signals locally and send the derived information to the back end once, reducing client-server data flow. In fact, a considerable power at rain-gauges is used by network communication: the typical current drain at 12 Vdc goes from 1 to 16 mA when not communicating to 17 to 28 mA during communication, to which we have to add the modem power consumption (3 Watt). The results presented in Table 3.3 assume that communication takes place for around 20% of the time, a rather conservative estimate. In total, we achieve a 81.69% saving using the 0th percentile, 90.84% using the 50th percentile, and 95.42% with the 75th percentile. This is not only important in cost reduction, but especially in prolonging the lifetime of batteries during power outages, which typically occur when the data are most needed; that is, during extreme weather.



(a) (b)

Figure 3.12: Heatmap of 21577 rainfall-related complaints between 2004 and 2001in Rotterdam. Figure 3.12a and Figure 3.12b are general and close-up views of the heatmap, which has a resolution of 1Ha and was built with 1Km ranges. It represents an estimation of the historic amount of complaints within 1 km range at every Ha of the modeled surface, and allows us to easily visualize the areas with higher complaints incidence. In fact, the incidence of complaints is not evenly distributed in the 277 sq km of Rotterdam surface (3.12a). Central and Southern Rotterdam present higher amounts of complaints. In 3.12b the vulnerable area of central Rotterdam is mostly covered by 11 radar cells.

The proposed system has the ability to improve the current spatial resolution as well. Current rainfall measurements have a maximum spatial resolution of 1 sq km, and only raw imagery is available near-realtime. A corrected radar-rainfall takes a day (see Section 3.3.1.1). Regarding rain gauges, the spatial resolution is limited to the number of stations currently installed, and the confidence of the interpolations derived from their measurements. A spatial resolution enhancement can be achieved by smartly placing affordable rain gauges at critical locations in the city. Those critical sites can be identified by analyzing the areas that presented more complaints related to rainfall issues. Figure 3.12 presents a heatmap of complaints in Rotterdam, and shows that complaint incidence is particularly high in central and southern Rotterdam. Rainfall measurements with higher spatio-temporal resolution can be performed in such vulnerable areas by focusing the installation of additional weather sensors there. The architecture presented in this work enables an easy, plug and play installation of affordable sensors to support higher spatial resolution. While inexpensive sensors operate with lower precision, they are worth the tradeoff when the goal is enhancing spatial and temporal resolutions at vulnerable areas.

In a theoretical squared radar grid, made of 9 cells of 1 sq Km each, the installation of 25 rain gauges, separated 750 m from each other, would provide a spatial resolution

of 0.56 sq KM. This means an enhancement of 78% in the spatial detail of rainfall measurements. For the focus area in Figure 3.12b, 30 to 31 raingauges would be required to achieve such performance. Additionally, such detailed spatial data would be available real-time, in comparison with the day-after availability for corrected radar rainfall data from the KNMI radars.

Chapter 4

Privacy Issues in the Cellular Age

On April 20, 2011, U.K. researchers Alasdair Allan and Peter Warden caused a media frenzy by announcing their discovery of an iPhone file that contained time-stamped user-location data. A FAQ published by Apple and congressional testimony by Apple's vice president for software technology subsequently revealed that at least some of the initial concerns were groundless. Assuming Apple's anonymity-preservation techniques are adequate, Apple does not compile location traces for individual users, instead enlisting those users as data collectors in a worldwide exercise in crowdsourcing. Apple is creating a highly precise map of cell sites and access points in an effort to improve the speed and accuracy of its user-location estimates, thus providing more-refined location-based services. However, despite Apple's quick and thorough response, long-term issues remain [131].

Cellular telephony has always been a surveillance technology. Cellular networks are designed to track a phone's location in order to route incoming calls to the most appropriate cell tower, i.e. the one closest to the user. The more this technology evolves, the finer becomes location resolution. Moreover, many mobile devices now have some form of GPS capability, whether standalone or network-assisted. Subsequently to the widespread diffusion of this kind of technologies, service providers increasingly recognize that a much broader and lucrative range of location-based services could be provided.

Nowadays, a lot of smart services or social applications are strictly related to the disclosure of the user's location. It is enough to reason about location tags in social networks posts or cellular applications designed to provide the citizen with useful information about his surroundings (which often come with location-aware advertising). These kind of services were widely discussed in Section 2.0.5.

Through the correlation of fine-grain location data with publicly available information it is possible to derive a lot of personal information such as the location of the

user's home, the location of his friends' home, the places he often visits, a religious affiliation, some serious illness related to frequent visits to a hospital or similar facilities [131]. Databases containing such information represent a threat to individual security and privacy.

We can sum up saying that location-based data open a new dimension in personal information management and dramatically extend privacy threats for end-users.

In the following, the discussion focuses on some author's proposals related to this subject. First, the threats to users' privacy are discussed in relation with common smart services like automated public transport ticketing systems and electronic identity documents. The lack of location privacy in the public transport field is discussed along with some novel techniques and best practices proposed to preserve user's privacy [7].

Finally, the author proposes a novel technique to preserve both users and providers privacy in location-aware services [96].

4.1 Privacy in location-aware public transport

In 2013, East Japan Railway (JR East), the largest rail company in the country, announced the intention to sell to Hitachi corporation a large dataset of its passengers' travel histories [38]. This information has been gathered by JR East through its proprietary e-ticketing system, *Suica*. The company plans to anonymize these data by replacing sensitive information, such as names and addresses of card owners, with anonymous ID's. But is this enough to protect users' identities, and therefore their privacy? Historically, public releases of anonymized personal information have often proved to be dangerous for the privacy of the people that information concerned. In 2006, America On Line (AOL) released anonymized data regarding the search queries of millions of users of its web search engine. Even if the IP addresses of the users were replaced by anonymous identifiers, researchers and even journalists had little trouble finding the real names of people corresponding to the anonymous ID's, as proved by the famous case of user #4417749, presented in a New York Times article [9]. In this thesis it is proved how the disclosure of travel histories can be equally dangerous, as travel data contain a great deal of information about the user, even when her real identity is concealed by means of anonymization.

The use of electronic tickets, usually smart cards, in public transportation networks has a number of potential benefits, both for the users and the provider of the transportation service. Often, their introduction also coincides with a more general

modernization of the transportation infrastructure. Modern networks usually integrate a positioning system (GPS) for monitoring the movements of buses and trams, backed by a constant Internet connection to a central control infrastructure. Enabling location-awareness allows, for instance, to display real time information and waiting times for each line on the provider's website or on information screens at bus stops and represent a value-added *city-to-citizen* service in the smart urban ecosystem [23] as discussed in Chapter 2. Internet communication between vehicles and a central server can also be used to signal traffic congestion or unexpected issues efficiently, in both directions. These innovations help in making our cities smarter and greener, by improving the quality and reliability of the public transport service.

However, the technology enabling these features also generates an unprecedented amount of information regarding user movements. And after such information is generated, the tendency among public transportation companies is to record it, rather than discard it when it exhausted its original goal. In this thesis privacy implications of such a large amount of data are discussed, and the potential consequences of its disclosure are analyzed. As electronic tickets are generally characterized by a unique ID, and all trips are recorded, the information stored in the information system of a public transport company is nothing less than a detailed log of each user's movements and therefore should be treated as sensitive information. However, in the following it is proved that even when personal information of the users are not stored in the system - or are anonymized - the threat to privacy remains. In fact, combining the data in the transportation company database with other publicly available source of information can ultimately be enough to identify a specific user, even in the case of anonymous tickets.

As stated by Diaz and Gurses in [32], it is often very difficult for individuals to know if their personal data are used according to the privacy terms and conditions provided by the company that holds them. Actually, when a company collects a large amount of data such as the user's position (like a public transportation one does), the problem of avoiding malicious surveillance, profiling or manipulation is better addressed pursuing a *Privacy as Confidentiality* paradigm, i.e. providing data anonymity by designing the appropriate protocols and procedures as hard-coded in the system itself. The privacy-friendliness of the infrastructure, if correctly implemented, does not necessarily hinder the business model, as shown in [74].

Many of the recent studies on the subject discuss the privacy issues introduced by the use of electronic documents from the point of view of achieving security against external attackers. A typical attacker is therefore some unauthorized person trying to

monitor the movements of a user, for instance by accessing the records of those movements stored on the ticket itself (usually a smartcard). For this reason, the studies usually conclude that no sensible information should be kept within the smartcard for longer than it is actually required for the correct functioning of the system. This is the case of [8], where the authors discovered, through an analysis of the Mobib smartcard (the transport pass used by the public network of the city of Brussels, Belgium) the presence of unneeded information that could expose users to privacy threats.

Let us consider a RFID-based public transportation information system: usually in this scenario the tag identifier is sent from the reader to the back-end server with some kind of encryption [5]. This is often achieved applying a collision resistant hash function to the identifier. Unfortunately, this does not prevent from coupling different stamps belonging to the same tag (i.e. the same user) and lacks anonimity [105]. In [47] the authors proposed an integrated model involving anonymous credential and e-cash systems in order to obtain an anonymous payment system for public transit; however they assume to dispose of devices capable of heavy and intensive computation, not really reasonable for available RFID systems, which use cheap tags and readers.

In the following a focus on anonymous, disposable 10-ride electronic tickets for public transportation is posed. Such tickets can be generally bought anonymously through resellers or automated machines and, while they are identified through a unique ID, they do not carry any information on the owner's identity. Thus, these tickets are perceived as the most privacy-friendly by the users while, at the same time, retaining some of the advantages of personal travel passes, such as a lower cost per ride than single-ride tickets, and the ability to be used multiple times. The choice of anonymous tickets allows to evaluate the potential effects of disclosure of travel histories to third parties, even when limited to a small number of rides and anonymized by removing personal information.

In this thesis the author presents the case study of a real, city-wide public transport network in Italy [7]. By analyzing and decoding the tickets issued by the company, the information collected during their use is inferred. This knowledge is used to show that even anonymized and numerically limited travel histories are indeed enough to profile users with a great depth of detail. It is also shown that careful elaboration of these data, and comparison with other publicly available sources of information ultimately allows to find matching patterns and to statistically identify the user as belonging to a small, well-defined group. Empirical evidence produced

by analyzing this case study proves that simple anonymization of the travel histories of public transportation users is not sufficient to protect their privacy, and therefore suggests caution in the disclosure or trade of such data without the informed consent of the users themselves. In order to address this issue, a set of recommendations for the design and management of the information systems of transportation companies is proposed. These solutions are both privacy-preserving and cost-effective, as they reduce the overhead in communication and storage of travel data to the information system, while avoiding costly renovations of current infrastructures.

In the following a specific case study (the Italian city of Cesena and its public transport system) is discussed and the potential information disclosure for a specific set of users (university students) is analyzed. However, the problem here highlighted is indeed common to other cities and countries. If in this work the author uses public information on students' classes and housing, the same result could be achieved, for instance, using phone directories. Overall, the aim of this proposal is not to prove a flaw in the design of a specific e-ticketing system, but rather to show how the disclosure or sale of location-aware information, such as travel histories, even when anonymous (or anonymized) could become dangerous to the privacy of the concerned individuals: in fact, when data are combined with other sources of information, the presumed anonymity disappears.

As discussed by [32], it is often difficult for individuals to know how their personal data are used by companies that hold them. While Diaz and Gurses mostly focus on sensitive data as defined by European Union regulations, their reasoning is equally valid when applied to companies collecting location data such as travel histories, as in the case of public transportation companies. In fact, users are often unaware of the risks of malicious surveillance, profiling or manipulation they are exposed to [8]. The security of this information is therefore best assured adopting the *Privacy by Design* paradigm, i.e. providing data anonymity by designing the appropriate protocols and procedures as hard-coded in the system itself [32]. The privacy-friendliness of the infrastructure, if correctly implemented, does not necessarily hinder the business model [74]. In the case of public transportation, electronic tickets raise privacy concerns for their ability to track users. Recent studies on the subject discuss this issue from the point of view of security against external attackers [5, 8, 47, 105]. A typical attacker is therefore some unauthorized person trying to monitor the movements of a victim, for instance by accessing the records of those movements stored on the ticket itself (usually a smartcard). For this reason, the studies usually conclude that no sensible information should be kept within the smartcard for longer than it is actually

required for the correct functioning of the system. This is the case of [8], where the authors discovered, through an analysis of the Mobib smartcard (the public transport pass used in the city of Brussels, Belgium) the presence of unneeded information that could expose users to privacy threats. In this thesis, instead, the author is interested in privacy with respect to the company providing the transportation service. In a typical scenario of an RFID-based ticketing system, the smart card ticket is read on the vehicle in order to learn its unique identifier, which is then sent from the reader to the central server encrypted [5], usually by applying a collision resistant hash function to the identifier. Unfortunately, this allows different stamps to be associated with the same user and therefore permits tracking [105]. In [59] the authors focus on electronic cash payments and bill processing in the e-ticketing scenario and discuss how to achieve a privacy preserving billing system based on asymmetric key encryption while in [99] a simple billing mechanism designed to avoid privacy leaks is proposed. Basically, it enables the public transport company not to collect the starting place and the ending one in order to compute the journey cost. Security and privacy issues related to *Near Field Communication*, a very common technology used for mobile-payments on public transports, are discussed by [108]. In [124], the use of anonymous databases for collecting user movements is discussed. The authors show, from a theoretical point of view, that anonymity alone is not enough to protect users' privacy. In this thesis their intuition is confirmed through the discussion of a real-world case study. Further, a set of plug-in privacy enhancements for existing information systems is proposed.

4.1.1 Passengers information management in public transport

Modern transportation systems integrate monitoring technology, both location based (GPS) and users based (smartcard tickets), that generates real time information on the current status of the transportation network. This constant flow of information is generally elaborated in real time, but is also stored and kept, usually indefinitely, by the relevant actors.

Passengers data collected by transportation companies are used for a number of purposes, with the most obvious one being ensuring that passengers pay the bus fare. In general, this can be achieved by verifying the authenticity of the ticket and its validity for the current ride, and this kind of checks is only performed during the ride itself. In modern ticketing systems, however, information on bus rides is often transmitted to a central information system, where it is stored for an indefinite period

of time. This allows the company to gather statistics and useful contextualized data that can be used later to monitor usage, propose modifications to the bus network and adjust frequency, fares, etc [30]. But, as the JR East/Hitachi case shows, these data are also of increasing interest to external actors, and therefore have an inherent monetary value. If internal policies or existing legislation do not prevent this, it is realistic to predict the emergence of a number of cases in which this information will be old to external (and possibly foreign) entities.

Table 4.1 shows the main information units that are commonly kept in information systems of public transport companies [109].They are divided in two main subgroups, namely statistics and pricing.

This large amount of recorded data poses an inherent threat to the privacy of the citizens. This threat is not necessarily linked to the increased integration of electronic identification mechanisms. It is in fact the sheer amount of data that enables tracking and tracing of the citizens: when everything a person does is registered, identifying that person among a group is a trivial task. In the following, it is proposed an example of how this is possible even when the available data are apparently limited, such as in the case of anonymous 10-ride tickets for the public transport network of a small city.

4.1.2 Case study: the Cesena bus network

The Italian city of Cesena is a small-sized town (less than 100.000 inhabitants) hosting a university campus. The campus, part of the Università di Bologna, offers five different majors (computer science, electrical and bio-medical engineering, architecture, agricultural sciences and psychology) to around 4.000 students (Table 4.2). The city is served by a bus network, provided by the regional public transport company Start Romagna, counting 6 city lines and 13 connections to nearby cities. The local population of university students is one of the main customers of the transport network, and benefits from specific fare discounts.

After a recent renovation of the bus network and the introduction of electronic tickets (called MiMuovo), an intelligent information system has been put in place by the transportation company: buses collect information about the passengers when reading the electronic tickets and send this information, along with their current position (learned from a GPS receiver) to the central system. The central system stores these data and updates both the company website and digital information screens at bus stops with real-time information on waiting times for the different bus lines. This technological system offers various advantages to the users: RFID tickets

	Information	Time requirements	Privacy implications
Pricing	**P.1** A sorted list of each stamp performed by each single identified user, in order to compute his fare (for travels with connections, composed of multiple stamps). **P.2** Discounts according to well-defined categories (students, elderly, people with disabilities . . .). **P.3** Zone-based or stop-based discounts (special tickets, such as airport shuttles, special events/destinations passes etc.).	Pricing information should be stored for billing purposes only and thus they should be deleted after each invoice issuance.	For billing purposes personal information of the user might be needed. However, the short-lived time required to perform these operations reduces the privacy implications. Moreover the user is generally well-aware of the company disposal of his personal information required for billing (address, credit card data, . . .) as he provided them himself.
Statistics	**S.1** Total amount of passengers for each ride of a line, in order to monitor the line workload. **S.2** A sorted list of each stamp performed by users during a ride, in order to understand anonymous patterns in user' habits. **S.3** A sorted list of each stamp performed by each single user during a single day in order to understand one-way and return patterns in users' habits. **S.4** Total amount of journeys related to a single ticket, in order to monitor the workload of a long-term (monthly, yearly) pass.	This information needs to be stored for a long time as they are used to perform statistical analysis, even in a long-term year-over-year comparisons.	Trip information collected by transport companies are less sensitive than billing data, but more invasive for mainly three reasons. First, the user might not be aware of the recording and storing of this information, contrary to the case of billing data. Second, the information needs to be stored for significantly longer periods of time in order to be useful. Finally, travel histories contain location information, which imply the users habits and the places he regularly visits, along with time and frequencies of the movements. These are potentially more invasive to the privacy of the users than mere financial records.

Table 4.1: Information typically handled by public transport companies.

Major	1st	2nd	3rd	Tot.
Agriculture	221	151	207	579
Architecture	133	127	167	427
Psychology	506	606	441	1553
Computer Science	229	140	190	559
Engineering	329	299	407	1035
Total	**1453**	**1350**	**1458**	**4261**

Table 4.2: Students enrolled in Bachelor's degrees offered at the Cesena Campus, by year (survey for the academic year 2012-13).

are faster to stamp, and real-time information on buses, also available through a mobile application, is precious for avoiding long waits at a bus stop. However, the amount of data collected by the system exposes the users to tracking. Personal tickets, such as monthly and yearly passes, are directly linked to a user, and both a government issued ID and the university card have to be showed upon purchase. Since each use is registered by the system, they allow the creation of a complete profile of the user's movements over the years. Many users may not find this problematic, but others are more concerned about their privacy.

A privacy-concerned user can however opt for a different kind of ticket offered by the transportation company: an anonymous 10-ride pass. This ticket retains some of the advantages of monthly and yearly passes, such as being less expensive than a single-ride ticket (especially with a student discount), but can be bought anonymously at newsstands and convenience stores. The user might therefore reasonably expect tracking to be impossible, and consequently a better privacy protection.

4.1.2.1 Analyzing an anonymous bus ticket

A 10-ride disposable MiMuovo ticket contains a Mifare Ultralight contactless integrated circuit (IC). Such an IC belongs to the cheapest memory-based contactless technologies commercially available. The ticket is nothing more than a 64-byte memory (EEPROM) readable and writable remotely, through a high-frequency radio channel. The IC does not contain any mechanism to protect the access to the memory or to ensure the confidentiality of the stored data. A security mechanism, though, allows anyone to lock memory areas such that write operations on these areas are no longer functional afterward. However, read-access to the memory cannot be prevented.

The interface of the IC relies on the widely used ISO-14443 standard, and the memory access is compliant with the well-known ISO-7816 standard. Reading a Mifare Ultralight IC is consequently quite easy using commercial readers and softwares. For example, an NFC-compliant smartphone with an appropriate application (e.g., *Tag Info*) is enough to read the memory of a MiMuovo ticket.

Reading the memory actually means "obtaining a verbatim copy of the memory". The content of the memory is not encrypted but it must be decoded in order to retrieve the intelligible information. This can be easily done because the public transportation tickets usually contain common information, e.g., date, remaining trips, bus line, identity, location, and the encoding method is generally based on public standards, e.g., ISO-1545. Performing a differential analysis is usually enough to complete the full decoding of the ticket: in such an approach, the ticket is punched a

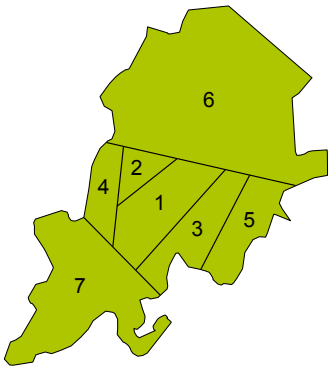


Figure 4.1: Cesena district partitioned into seven areas, according to which bus line each area is covered by. The bus and train station is in area 1.

few times, taking care that only one information (date, location, bus line,...) varies at a time. This allows identifying the fields encoded in the memory. Doing so, one can retrieve the information stored in a MiMuovo ticket. The memory actually contains 32 bytes of technical data whose write-access is partially locked, and 32 additional bytes that are freely modifiable. The latter bytes contain two 16-byte fields such that they are refreshed cyclically when the ticket is punched. Each field essentially contains the journey identifier, validation date and time, connection time, and also the geographical zone. The zone is a particularly sensitive information in terms of privacy.

4.1.2.2 Breaking anonymity

Let us analyze the information collected by an anonymous ticket in its geographical context. In particular, we are interested in the topology of the bus network serving Cesena. As it is usually the case in small cities, Cesena is served by a number of bus lines, all of which have the central bus station as a starting point. From there, different lines branch out to reach different areas of the city.

In general, apart from the city center, no two lines cover the same area. This allows us to roughly divide the city in seven zones, according to which line they are served by. Zone number 1 is the city center, where the bus and train station are located, while zones numbered from 2 to 5 are neighbors covered by different bus lines. Zones 6 and 7 represent instead suburban locations served by provincial buses.

As most Italian city campuses, university sites and lecture halls are spread around the city, with each major having a different location. The proposed partition of the Cesena district also reflects this distribution: in particular, Psychology and Computer Science are located in two different buildings both in zone 1, Engineering is in zone

4, Architecture in 2, while Agriculture is outside of the city boundaries and therefore served by buses of zone 6. University buildings are usually best reached from one specific bus stop. Therefore, users of student-discounted tickets can be easily divided according to their major when a significant number of stamps are made at one of those bus stops. Moreover, the timestamps of those stamps can help an observer identify the year of study of a specific student/bus user: in fact, each class schedule is different depending on the year a student is enrolled in (in Figure 4.2, the bachelor in Computer Science).

If the student in question is lodged in university housing (which usually hosts students coming from outside the region, the most likely ones to use public transportation) profiling him based on his use of the bus pass can be even more successful. Places in the dormitories, located in zone 3, are assigned through a public selection whose results are available online. The concurrent analysis of this information, all of which is publicly available, with data collected through bus tickets by the transportation company can be enough to disclose the identity of the owner of a bus pass even when the pass is supposed to be anonymous, as in the case of 10-ride tickets.

In fact, according to the most recent report on public transportation published by the Italian national statistics institute, ISTAT, the number of university students regularly using public transportation in the city where their campus is located is 31% [53]. This figure is consistent with publicly available data on discounted tickets issued by Start Romagna for the city of Cesena and, combined with the numbers in Table

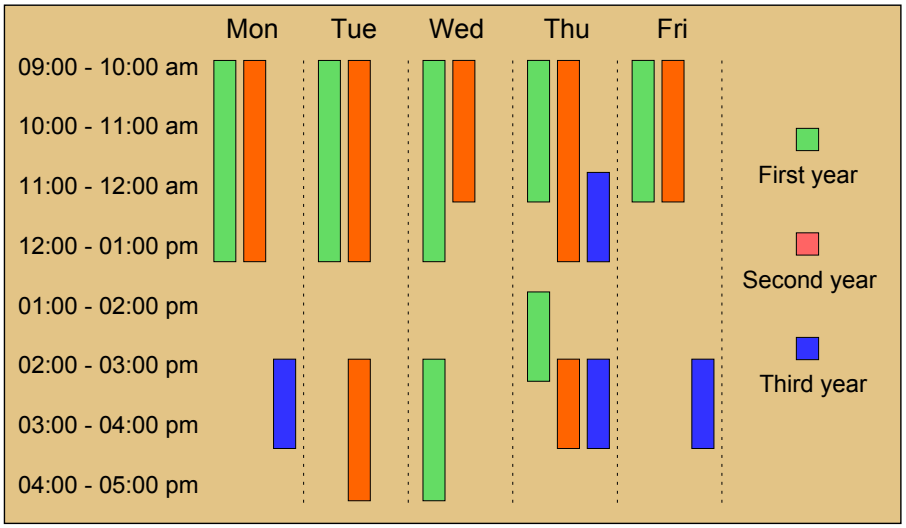


Figure 4.2: Time and location of classes make years distinguishable. Here, the schedule for the major in Computer Science.

4.2, means that groups of users sharing the same characteristics (major, year of study, housing, ...) are composed of a handful of students only.

As explained in Section 4.1.3 it is to be noticed that sometimes it is not possible to hide ore screen certain types of data. And as explained herein, this policy leads to serious threats for the user privacy.

	Aggregate	**Minimal field set**	**Privacy implications**
S1	Passengers for each ride of a line	*Ride-id*, *Line-id*	Uncontextualized trip information. No threat.
S2	Routes (first stamp and changes)	*Ride-id*, *Line-id*, *Timestamp*, *Stop-id**	Exact trip information, but unlinked to the user. Reduced threat.
S3	Coupled-routes (returning ticket)	*Ride-id*, *Line-id*, *Timestamp*, *Stop-id**, *Ticket-id*	Aggregate requires all sensitive information. The ticket-id must therefore be removed or encrypted to prevent linking with other records.
S4	Journeys per ticket	*Ticket-id*	Number of rides counted. Implies the frequency a user travels.

Table 4.3: Aggregates typically used by public transport companies. Each aggregate needs a set of atomic data in order to be computed. Note that as public transport companies usually know at any time the exact position of each controlled vehicle (thanks to GPS) the field *Stop-id* can be inferred from the field *Timestamp*.

4.1.3 Laws pertaining public information

In this work the author shows ways of de-anonymizing travel histories by comparing them to other sources of information. In order to show the viability of this approach, only publicly available information are used. The presented case study focuses on Cesena's university students: therefore, information from the local university dormitories and housing directories are used. In the following, some legal references is provided. It shows how it is, in general, mandatory to maintain this personal information publicly accessible: this is due to transparency policies for applicants in merit rankings.

According to the Italian law D.P.R. 09.05.1994 n. 487 (published on the official gazette n. 185 on August 9, 1994), each ranking list related to a public competition must be published and accessible to the public. This is the case for instance of subsidized housing for students or public housing for disadvantaged people. More generally, it is commonly stated in law that the protection of personal data of an individual shall not apply when the exposure of such information is required by law

itself for reasons of transparency and public access to information held by authorities. In practice, some public lists of various kinds will always be exposed, due to the need of balancing state responsibilities in terms of transparency and human rights in terms of privacy protection.

4.1.4 Plug-in privacy enhancements

As most of the privacy threats concerning pricing information can be overcome deleting related data after each invoice issuance, in the following a focus on raw data used for monitoring, statistics and similar purposes is posed. The main aggregate information needed by public transportation providers in these cases are summarized in Table 4.3. Aggregates are basic information needed to compute meaningful statistics.

For example, if the bus company wants to analyze if the frequency of a bus line is adequate to demand, aggregate S1 is to be used. For this kind of aggregate it is enough to keep track of the bus line identifier and the specific ride identifier. Aggregates S2 and S3 are used instead in monitoring complex usage patterns, such as frequency and location of stopovers between bus lines during a single ride (S2) and behavior of the userbase on an outbound and then inbound journey (called coupled routes, S3). An abundance of the same entries on aggregate S2 may indicate that a direct link between one stop and another should be established. Aggregate S3 can instead detect patterns in the behaviors of the users. For instance, it lets the company understand the route a user usually takes during his working (or studying) day, allowing to study one-way and return patterns. In order to do that, the ticket identifier is also required. Finally, storing the ticket identifier alone is enough to know the usage load of a long-term subscription.

Recording this information, however, realizes the threat to privacy discussed herein. In [114] the authors show that consumers usually consider corporations responsible for any inappropriate use of personal information. The organization management should therefore enforce adequate information privacy policies and promote an information systems emphasizing built-in privacy preserving features.

In fact, a well designed system architecture can prevent privacy threats while allowing the same computations to be performed. In Figure 4.3 three different approaches to the information exchange between the local recording point (e.g. the bus) and the central storage system are proposed. In the first model, the validation machine sends to the central server the field set required to compute the discussed aggregates at the time of the ticket punch. Data are transmitted in a single atom with the identifier unencrypted. This basic and unfortunately commonly used model does

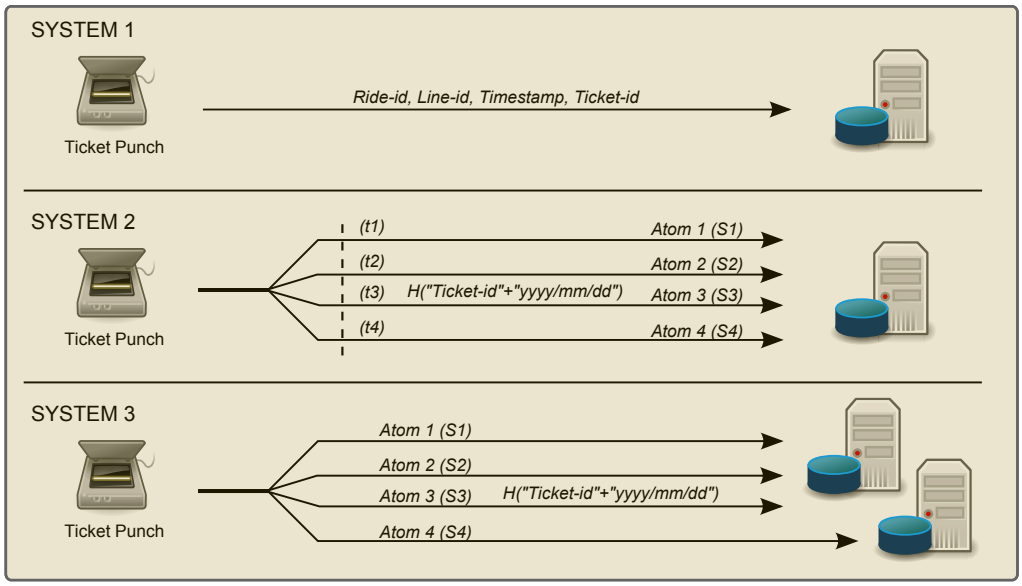


Figure 4.3: Different models for raw information management in the public transport scenario as discussed in Section 4.1.4.

nothing to prevent the potential privacy breaches discussed above, as the identifier for the ticket or pass is directly linked to timestamps, bus lines/stops and so on.

The second system model relies on a more advanced and privacy friendly approach. Here the same data are transmitted to the same remote storage, but in four different atoms. Each atom contains only the minimal data set required to compute a single aggregate. Delivering these atoms individually and at different times and encrypting the *ticket-id* (for aggregate S3) actively breaks the link between the identifier and other information and therefore enhances privacy. Note that, as aggregate S3 is intended to monitor daily patterns, the encrypted ticket identifier has to stay the same only for the duration of the day. Therefore, it would be possible to apply a one-way hash function on the string `"ticket-id"+"yyyy/mm/dd"`, producing unique, one-day encrypted identifiers for each different ticket.

Following these design principles and assuming that the hardware of ticket punch machines can be trusted, the *privacy by design* paradigm is achieved, as the privacy preserving properties are hard-coded into the system itself and the public transportation company only receives data that are natively anonymized. At the same time, the company is still able to compute meaningful statistics. For example, receiving the minimal field set related to *Atom 1* (*Ride-id*, *Line-id*) separately from other field sets, does not prevent computing the aggregate *Passengers for each ride of a line*, but prevents performing an implicit time-based analysis in order to link this information to information of other atoms.

Finally, the third system model reaches a similar privacy enhancement by transmitting atoms to different remote servers, controlled by different business units within the company or by different companies. In fact, as described by [56], distributing data among autonomous and independent sites provides protection for individual data. The assumption here is that different controllers do not collude with each other by sharing information. In our case, performing a vertical partitioning of data concerning aggregates S1, S2 and S3 (and encrypting the *ticket-id* in S3 at the server side) on one side and aggregate S4 on the other, prevents profile reconstruction by a `JOIN` operation on the atoms timestamps or other sensitive fields which could disclose the ticket identifier relative to specific routes, timings and positions. To deploy such a system on an existing platform, it is sufficient to partition the previously collected data and to start recording new ones in separate business units. The most common choice would be to deploy a relational database via a hosting service. In order to do that it would be advisable to consider a database instance with a read replica designed for disaster recovery. The database should be able to support write-intensive policies and should be up and running 24 hours a day. Focusing on *Atom 4* and considering a 64 byte record required to store a single atom, a municipality reporting 1 million validations per day would produce approximately 2 GB data monthly. However, this model does not reach the *privacy by design* paradigm, as the level of privacy achieved is strictly related to the trust implied in the parties managing the information, for the whole lifetime of the system.

4.1.4.1 Costs and potential disadvantages of the proposed solutions

The proposed systems introduce several benefits concerning user's privacy, but also come with some potential disadvantages in terms of costs of deployment and data processing limitations. In order to deploy the second system proposed in Figure 4.3, it is mandatory to embed into the ticket punch machines a *jitter* that introduces randomness in atom delivery times and an algorithm for computing a secure hash function. These changes usually require a firmware upgrade. However, should the currently used processor not be capable of performing such tasks, the hardware upgrade costs are also to be considered. Depending on the complexity of the modifications required by the ticket punches and the data processing strategies applied, an implementation cost should also be added. For these reasons, the third system is preferable when privacy needs to be achieved in an already existing system, as it does not imply the costs associated with the upgrade of validating machines already installed on vehicles. In this case, the only measure involving ticket punches in an existing system would be a

firmware update in order to send *Atom 4* to a different server. The most meaningful cost introduced in this model is data hosting, a service we assume is provided by a third company. In general, these costs depend on the current status of the system used by the company. For instance, buying new compliant ticket punches could be less cost effective than to upgrade existing ones. These considerations are summed up in Table 4.4, where two different scenarios are considered: in the first one, the transport company is about to deploy an entirely new system, while in the second one the company decides to upgrade the existing system instead.

	Build from scratch	**System upgrade**
System 2	System design	System design, hardware upgrade or purchase, firmware upgrade, server-side processing update
System 3	Data hosting	Slight firmware upgrade, data hosting, server-side processing update

Table 4.4: Costs introduced adopting the privacy preserving systems proposed in Section 4.1.4. In the *build from scratch* case we exclude the purchase of ticket punches as they should be buyed even if a traditional system would be adopted.

Concerning limitations on data processing, both systems preserve the ability to compute all the aggregates proposed in Table 4.3. Problems may arise if the public transport company attempts to compare routes belonging to the same anonymous ticket on a basis of more than one day, in order to study the user's movements and learn more complex patterns. This is however exactly the reason transport companies should introduce such systems: to guarantee users that their data are not used for malicious tracking and increase customers' trust in public transport systems, and, consequently, encourage more widespread usage of public transport itself.

4.2 Privacy and security in electronic identity documents

The widespread usage of electronic identity documents represents a serious threat for citizens' privacy.

A biometric passport, also known as an e-passport, ePassport or a digital passport, is a combined paper and electronic passport that contains biometric information that can be used to authenticate the identity of travelers [50]. It uses contactless smart card technology, including a microprocessor chip (computer chip) and antenna (for

both power to the chip and communication) embedded in the front or back cover, or center page, of the passport. Document and chip characteristics are documented in the International Civil Aviation Organization's (ICAO) Doc 9303. The passport's critical information is both printed on the data page of the passport and stored in the chip. Public Key Infrastructure (PKI) is used to authenticate the data stored electronically in the passport chip making it expensive and difficult to forge when all security mechanisms are fully and correctly implemented.

The currently standardized biometrics used for this type of identification system are facial recognition, fingerprint recognition, and iris recognition. These were adopted after assessment of several different kinds of biometrics including retinal scan. The ICAO defines the biometric file formats and communication protocols to be used in passports. Only the digital image (usually in JPEG or JPEG2000 format) of each biometric feature is actually stored in the chip. The comparison of biometric features is performed outside the passport chip by electronic border control systems (e-borders). To store biometric data on the contactless chip, it includes a minimum of 32 kilobytes of EEPROM storage memory, and runs on an interface in accordance with the ISO/IEC 14443 international standard, amongst others. These standards intend interoperability between different countries and different manufacturers of passport books.

Concerning electronic identity documents, european passports planned to have digital imaging and fingerprint scan biometrics placed on the RFID chip. This combination of the biometrics aims to create an unrivaled level of security and protection against fraudulent identification papers. Technical specifications for the new passports can be found in [50].

For instance, in [83, 18] the authors show that one of the protocols provided by the electronic passport built under ICAO specifications could be used to keep a digitally signed user trace, i.e. a non-repudiable trace of his movements through country borders, airports and so forth.

The author also found a form of attack that could be used to steal a passport owner identity. This attack represents a considerable threat to a citizen security and privacy and is widely discussed in [22]. As the discussion relies on very specific arguments strictly related to ICAO guidelines and technical reports from European Union and other partners of the e-Passport project, the author considers it inappropriate to entirely include it within this thesis.

4.3 Enabling privacy preservation in location-aware services

The ability to detect movements and exact position of the users in outdoor environments has become widespread since the introduction of smartphones equipped with positioning systems such as a GPS receiver.

The ability to track a user's position raises however deep privacy concerns, due to the sensitive nature of location information. In fact, a number of potentially sensitive professional and personal information about an individual can be inferred knowing only her presence at specific places and times [7, 13]. Even anonymized position data sets (not containing name, phone number or other obvious references to the person) do not prevent precise identification of the user: in fact, just four mobility traces may be enough to identify her [31]. The more users disclose their data, the more providers are able to profile them in an accurate way. Smartphones and location-aware services are now an integral part of our everyday life, but it is reasonable to predict that in the coming years users will demand privacy safeguards for their location information [67, 135]. The real challenge is therefore how to protect the privacy of the user without losing the ability to deliver services based on her location [97].

A common application scenario of location-based services requires the service provider to learn when the user is close to some sensitive or interesting locations. This is the case, for instance, of "around-me" applications or security and military systems [23]. If we add to this scenario the requirement of the user position to stay private, the problem becomes an interesting and fundamental research question. In literature a similar problem, known as *private proximity testing* has been studied: Alice can test if she is close to Bob without either party revealing any other information about their location [89]. Narayanan et al. proposed a solution based on location tags (features of the physical environment) and relying on Facebook for the exchange of public keys [89]. His protocol was later improved in efficiency by Saldamli et al. [107]. Location tags and proximity tests are also used in [57], as a way of providing local authentication, while [138] presents a secure handshake for communication between the two actors in proximity. The security of the basic proximity testing protocol has been further improved in [90]. In [123], Tonicelli et al. propose a solution for proximity testing based on pre-distributed data, secure in the Universal Composability framework. Finally, the problem of checking the proximity in a specific time is addressed in [119].

In the following the focus is not posed on proximity testing, but on a broader and more general problem: testing in a private manner whether a user is within one of a set of areas of arbitrary size and shape. By solving this problem and applying an intelligent conformation of areas, we can also solve the proximity testing problem (for one or multiple points simultaneously), and we are actually able to identify with some precision the distance of the user from the point of interest. Given the conceptual similarity of our problem with membership testing in sets, the proposed solution is based on a novel modification of Bloom Filters (BF). Bloom filters are a compact data structure that allows to compute whether an element is a member of the set the filter has been built upon, without knowledge of the set itself [11]. Bloom filters have already been used in privacy-preservation protocols, and they are particularly suited to be used in conjunction with the homomorphic properties of certain public key encryption schemes [60].

The proposed modification of Bloom filters is aimed at managing location information. Two private positioning protocols for privacy-preserving location-aware applications are presented as well. The novel variant of Bloom filters introduced herein, called *Spatial Bloom Filter* (SBF), is specifically designed to deal with location information. Similarly to the classic Bloom filters, SBFs are also well suited to be used in privacy preserving applications. This feature is proved by presenting two protocols for private positioning. The first protocol is based on a two-party setting, where communication happens directly between the user of a location-based service and the service provider. A more complex scenario is defined in the second protocol, that involves a three-party setting in which the service provider outsources to a third party the communication with the user. Both settings do not assume any trust between the different parties involved. The protocols allow secure computation of location-aware information, while keeping the position of the user private. The only information disclosed to the provider is the user's vicinity to specific points of interest or his presence within predefined areas. At the same time, the areas of interest are not disclosed to the user. Therefore, unlike other works on location privacy, which is usually discussed from the end-user point of view, the model proposed herein falls in the secure multi-party computation field, where both parties have an interest in keeping their information private. Military and government applications are just the most immediate examples of when location privacy represents a key-problem for the provider as well.

4.3.1 Bloom filters

A Bloom Filter (BF) is a data structure that represents a set of elements in a space-efficient manner. A BF generated for a specific set allows membership queries on the originating set without knowledge of the set itself. The BF always determines positively if an element is in the set, while elements outside the set are generally determined negatively, but with a probabilistic false positive error.

Definition 3 *We define a* Bloom filter $B(S)$ *representing a set* $S = \{a_1, \ldots, a_n\}$ *as the set*

$$B(S) = \bigcup_{a \in S, h \in H} h(a) \ , \tag{4.1}$$

where $H = \{h_1, \ldots, h_k\}$ *is a set of* k *hash functions such that each* $h_i \in H : \{0,1\}^* \rightarrow \{1, \ldots, m\}$*, that is, the hash functions take binary strings as input and output a random number uniformly chosen in* $\{1, \ldots, m\}$*.*

A Bloom filter $B(S)$ can be represented as a binary vector b composed of m bits, where the i-th bit

$$b[i] = \begin{cases} 1 & \text{if } i \in B(S) \\ 0 & \text{if } i \notin B(S) \end{cases} \ . \tag{4.2}$$

The bloom filter is built as follows. Initially all bits are set to 0. Then, for each element $a \in S$ and for each $h \in H$ we calculate $h(a) = i$, and set the corresponding i-th bit of b to 1. Thus, m bits are needed in order to store b.

We test an element a_u against b to determine membership in S, that is, we verify whether $a_u \in S$ if

$$\forall h \in H, b[h(a_u)] = 1 \ . \tag{4.3}$$

If any bit in b that corresponds to a value output by one of the hash functions for a_u is 0, then $a_u \notin S$. If, instead, all the hashes map to bits of value 1, then $a_u \in S$ minus a false positive probability p determined by the number n of elements in S, the number k of hash functions in H and the maximum possible value m output by the hash functions (equal to the binary length of b) as follows:

$$p = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-\frac{kn}{m}}\right)^k \ . \tag{4.4}$$

This small false positive probability is due to the potential collision of hashes evaluated on different inputs, resulting into all bits associated to an element outside the originating set having value 1. As such, it is determined largely by k: if k is sufficiently small for given m and n, the resulting b is sufficiently sparse and collisions are

infrequent. If we consider the approximation in (4.4), we can calculate the optimal number of hashes k as

$$\text{opt}\,(k) = \frac{m}{n} \ln 2 \ , \tag{4.5}$$

from which we can infer

$$m = \left\lceil -\frac{n \ln p}{(\ln 2)^2} \right\rceil \ . \tag{4.6}$$

However, the number of hashes also determines the number of bits read for membership queries, the number of bits written for adding elements to the filter, and the computational cost of calculating the hashes themselves. Therefore, in constrained settings, we may choose to use a less than optimal k, according to performance reasons, if the resulting p is considered sufficiently low for the specific application domain.

4.3.2 Cryptographic primitives

In part of the proposed construction the author uses the homomorphic properties of encryption schemes. In general, a cipher has homomorphic properties when it is possible to perform certain computations on a ciphertext without decrypting it and, therefore, without knowledge of the decryption key. In particular, an encryption scheme is *additively homomorphic* when a specific operation $\boxplus$ applied on two ciphertexts $(\text{Enc}\,(p_1)\,, \text{Enc}\,(p_2))$ decrypts to the sum of their corresponding plaintexts $(p_1 + p_2)$:

$$\text{Dec}\,(\text{Enc}\,(p_1) \boxplus \text{Enc}\,(p_2)) = p_1 + p_2 \ . \tag{4.7}$$

There is additive homomorphism also when an operation on a ciphertext and a plaintext results in the sum of the two plaintexts. We have instead *multiplicative homomorphism* between an encrypted plaintext and a plaintext when an operation $\boxdot$ results into the multiplication of the two plaintexts:

$$\text{Dec}\,(\text{Enc}\,(p_1) \boxdot p_2) = p_1 \cdot p_2 \ . \tag{4.8}$$

An example of encryption scheme that is both additively and multiplicatively homomorphic is the Paillier cryptosystem [95]. In this case, the product of two ciphertexts will decrypt to the sum of their corresponding plaintexts (additive property), while an encrypted plaintext raised to the power of another plaintext will decrypt to the product of the two plaintexts (multiplicative property).

Private Hadamard Product The Hadamard (or entrywise) product of two vectors, one binary (owned by Alice) and one composed of natural numbers (owned by Bob), is performed in a privacy-preserving manner by Algorithm 4.1. The algorithm is private with respect to the input vectors, and only reveals the product vector to Alice. The security of the algorithm is based on the encryption of Alice's vector using a public key encryption scheme that is multiplicative homomorphic for operation $\boxdot$.

Algorithm 4.1: Private Hadamard product of an encrypted binary vector for a cleartext vector of natural numbers

Input Alice: $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\mathbf{X} \in \{0,1\}^n$.
Input Bob: $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$, $\mathbf{Y} \in \mathbb{N}^n$.
Output Alice: $\mathbf{X} \cdot \mathbf{Y}$.

1 Alice generates a public and private key pair using a multiplicative homomorphic encryption scheme, and sends the public key to Bob.
2 Alice sends to Bob the ciphertext vector $\mathbf{E} = (\mathrm{Enc}(\mathbf{x}_1), \ldots, \mathrm{Enc}(\mathbf{x}_n))$.
3 Bob computes the vector $\mathbf{C} = (\mathrm{Enc}(\mathbf{x}_1) \boxdot \mathbf{y}_1, \ldots, \mathrm{Enc}(\mathbf{x}_n) \boxdot \mathbf{y}_n)$ and sends the result to Alice.
4 Alice uses her secret key to decrypt $\mathbf{C}$ and obtains $\mathbf{D} = \mathrm{Dec}(\mathbf{C}) = \mathbf{X} \cdot \mathbf{Y}$.

A more conservative version of the algorithm requires Bob to multiply a randomly chosen prime number p, larger than any $\mathbf{y} \in \mathbf{Y}$, to each value in the vector, before performing the homomorphic multiplication. Alice can then obtain $\mathbf{X} \cdot \mathbf{Y}$ by calculating p using any greatest common divisor algorithm.

In general, it is assumed that the parties participating in the proposed construction do not deviate from the protocol, but gather all available information in order to try to learn private information of other parties. We are, therefore, in the semi-honest setting.

Security Model It is assumed the parties are *honest-but-curious*, that is, the parties will follow the protocol but try to learn additional information about other parties private data.

4.3.3 Spatial representation

The construction presented herein is based on a novel variant of BFs aimed at managing location information. Since BFs are constructed over finite sets of elements, we need to represent location information – that is, a geographical position – as an element that is part of the finite and discrete set of all possible positions. Therefore,

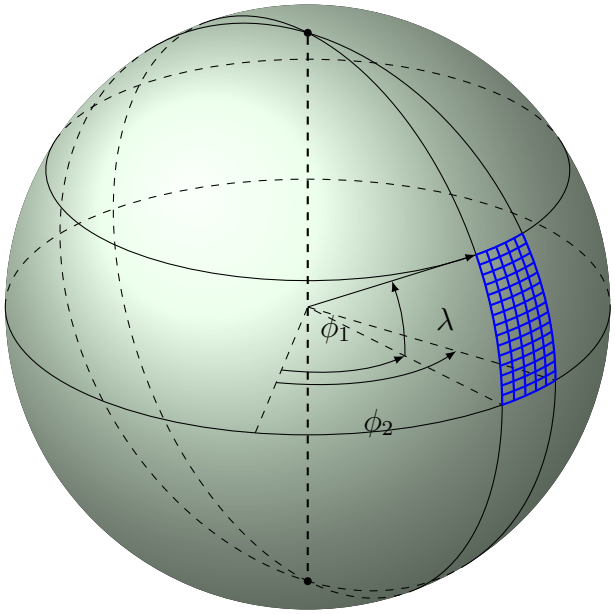


Figure 4.4: An example of the planet's surface and the grid plotted on it. ϕ_1 and ϕ_2 are longitude values while λ is a latitude value.

instead of considering a location as a point, we divide Earth's surface into a set of distinct areas, and we identify a position as the corresponding element in this set. Considering that it is possible set the dimension of such areas to an arbitrarily small size, there is no loss in the precision of the location information. In particular, this approach is not intended to obfuscate or partially hide an exact position: on the contrary, the main aim is to retain a precision as high as the one allowed by the location sensor used in the specific application.

The most natural spatial representation for Earth is the standard geographic coordinate system. In the geographic coordinate system every location on Earth can be specified by using a set of values, called coordinates. Standard coordinates are *latitude*, *longitude* and *elevation*. For the purposes of this work the focus is posed on longitude and latitude only, as the combination of these two components is enough to determine the position of any point on the planet (excluding elevation or depth). The whole Earth is divided with 180 parallels and 360 meridians; the plotted grid resulting on the surface is known as the *graticule* (Figure 4.4[1]).

Longitude (`lng`) and latitude (`lat`) can be stored and represented according to several formats. In the following the *decimal degrees plus/minus* format is adopted, where latitude is positive if it is north of the equator (negative otherwise), and longi-

[1]The author would like to acknowledge Marco Miani for the code used in producing this figure.

0.001 degrees	**lng**		**lat**	
equator	111.32 m	$\sim$	111.00 m	
23th parallel N/S	102.47 m	$\sim$	111.00 m	Cuba
45th parallel N/S	78.71 m	$\sim$	111.00 m	Italy
67th parallel N/S	43.50 m	$\sim$	111.00 m	Alaska

Table 4.5: Some reference values of accuracy using three decimal places for coordinate representation.

tude is positive if it is east of the prime meridian (negative otherwise); for instance, 31.456764°(lat) and −85.887734°(lng) are two possible values.

Using a fixed precision in longitude and latitude (that is, choosing a fixed number of decimal points for their values) allows to easily divide the planet's surface into a discrete grid. Since meridians get closer as they converge the poles, as can be seen in Figure 4.4, the portions of the Earth's surface defined by such a grid have varying areas depending on their position (Table 4.5). While the construction proposed in the following is not dependent on the size or shape of the regions, for simplicity in the discussion it is reasonable to approximate such portions to rectangles and assume they have the same area.

In actual applications, the precision in decimal points for longitude and latitude should reflect the expected error of the device or sensor used for learning the location information. The precision and accuracy of mobile devices in determining their geographic position were proved to vary considerably depending on the context (urban areas, rural areas, etc.) [130].

In a detailed experiment on the accuracy of GPS sensors installed on mobile devices, Blum et al. show that the location is reported with a precision varying from 10 to 60 meters, depending on the device orientation and type, and, in cities, on the surrounding buildings [12]. Hence, when designing a system based on mobile devices it would reasonable to consider regions with sides tens of meters long.

For the purpose of this work it is reasonable to consider the grid defined by longitude and latitude values with a precision of three decimal point places. This grid divides Earth's surface in a number of regions. The set of all regions is defined as follows.

Definition 4 *We define $\mathcal{E}$ as the set of all regions in which Earth's surface is divided by the grid defined by the circles (called parallels) of latitude distant multiples of*

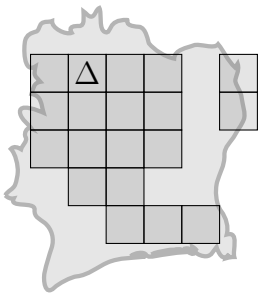


Figure 4.5: A sample area covered by an arbitrary grid.

0.001° *from the equator and the arcs (called meridians) of longitude distant multiples of* 0.001° *from the Prime Meridian.*

The sides (in meters) of a region of side 0.001 degrees in terms of longitude and latitude vary depending on its position on the globe. Table 4.5 contains some reference values.

4.3.3.1 Areas and Points of Interest (AoI & PoI)

The purpose of this proposal is to present a method able to preserve both user's and provider's privacy in location-aware applications. In the imagined scenario the provider of such an application wants to be notified of the presence of the user in one of a predefined set of areas of interest (AoI). The areas of interest are selected by the provider, and each is composed of an arbitrary number of regions in $\mathcal{E}$, defined above. An area may, for instance, represent a sensitive or interesting location for the purposes of the application. A number of concentric areas around a point of interest (PoI) can be used to detect the user's vicinity to the PoI. In the following two ways in which the set-based location information described above can be used to achieve this goal are presented. Both approaches are used in the following as strategies to select the areas of interest by the service provider, but is important to stress here that the proposed construction is independent of the strategy used, and therefore can accommodate any other set selection mechanism.

The first strategy to define a set of areas of interest follows naturally from the idea of detecting the presence/absence of a person in a given area. In order to do that, the provider of the service defines an area by selecting a subset of $\mathcal{E}$ (Figure 4.5). The regions in the area need not to be contiguous, and there is no limitations in shape or size of the area. The set containing all of these regions is defined as Δ (further discussed in Section 4.3.4).

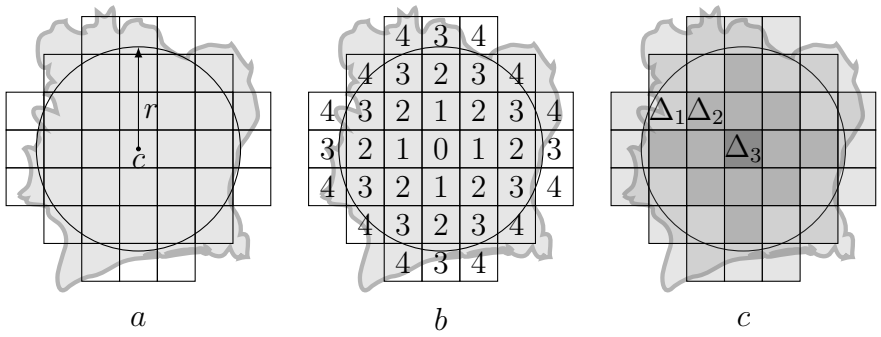


Figure 4.6: An example of the area coverage algorithm applied to a point of interest of interest. After defining the grid (a), the Manhattan distance from the center region is computed (b). Finally, each region is assigned to the right set (c). In this case, the maximum distance (σ) is 4, so we assign the regions belonging to the distance classes 4 and 3 to Δ_1, those belonging to the classes 2 and 1 to Δ_2 and the sole region belonging to the 0 class to the set Δ_3.

A second approach is instead to monitor the user by detecting his proximity to the area's center as he approaches it. This goal is achieved without knowing the user's exact location by defining several concentric areas around the point of interest to be monitored. In the example shown in Figure 4.6, three areas are used for this purpose, but this parameter can take any value deemed useful.

Let c be the center of the area (having coordinates lng_c, lat_c) and let r be the range we are interested to monitor users around the center itself. First of all a region is chosen such that it is the element of $\mathcal{E}$ that contains the point c. Then a number of adjacent elements (all belonging to $\mathcal{E}$) are added in order to form a grid, until the circle of center c and radius r is completely included in the grid, as shown in Figure 4.6a. Now let us label each region with its distance from the center region, using the standard *Manhattan distance*. Assume that σ is the maximum distance value in the generated grid; two cases need to be discussed. If $(\sigma + 1) \mod 3 = 0$, we assign to the set Δ_3 each region labeled from 0 to $q - 1$, where $q = (\sigma + 1)/3$. Similarly, we fill the set Δ_2 with each square labeled from q to $2q - 1$ and the set Δ_1 with each square labeled from $2q$ to σ. If 3 does not divide $\sigma + 1$ exactly (i.e. $(\sigma + 1) \mod 3 \neq 0$) some rounding is required; we could for instance assign the first remaining class to Δ_1 and the second optionally remaining class to Δ_2. In that case, given $q = \lfloor (\sigma + 1)/3 \rfloor$, the procedure can be formalized assigning each region labeled from 0 to $q - 1$ to the set Δ_3, each region labeled from q to $2q$ to the set Δ_2 and each region labeled from $2q + 1$ to σ to the set Δ_1.

4.3.4 Spatial Bloom Filter

After defining a spatial representation $\mathcal{E}$ of Earth's surface and providing a way to identify geographical areas (and points) as elements of a subset of $\mathcal{E}$, it is possible to use a set-based data structure like the Bloom filter to encode this information. However, the original definition of BF proves to be quite inefficient for this task, as it would be possible to encode only one area for each BF.

In the following the author defines a novel data structure called *Spatial Bloom Filter*. A spatial Bloom filter can be used, likewise the original BF, to perform membership queries on the originating set of elements without knowledge of the set itself. Contrary to the BF, however, a spatial Bloom filter can be constructed over multiple sets, and querying a spatial Bloom filter for an element returns the identifier of the specific set among all the originating sets in which the element is contained, minus a false positive probability (of assigning the element to the wrong set). Similarly to a classical BF, there is also a false positive probability that querying a SBF with an element outside the originating sets returns a positive result (wrongly assigning the element to one of the originating sets).

An important property of SBF is that the probability of false positives, that is, the probability that an element is wrongly recognized as belonging to a specific originating set, depends on the order in which the sets have been encoded in the filter: a false positive can occur either when an element outside the originating sets is recognized as being part of one, either when an element that is part of an originating set is recognized as being belonging to a different one (sets are disjoint). The latter case, however, can only happen if the wrongly recognized set has been encoded later than the actual originating set.

This fundamental property allows to define an order of priority for the different originating sets, thus reducing the error probability for elements (areas) deemed more important. Considering the strategies described in the previous section for selecting areas of interests, this property is particularly useful when using SBFs to store location information. In the example presented in Section 4.3.3.1, for instance, a set of three different areas $S = \{\Delta_1, \Delta_2, \Delta_3\}$ was used. Assuming the provider would prefer a more accurate monitoring of the area's central region, the highest label value (3) was assigned to the inner area. In the following the sets are generally considered as already ordered by priority, meaning that set Δ_2 is considered as having higher priority than Δ_1.

Definition 5 *Let $S = \{\Delta_1, \Delta_2, \ldots, \Delta_s\}$ be a set of areas of interest such that $\Delta_i \subseteq \mathcal{E}$ and S is a partition of the union set $\bar{S} = \bigcup_{\Delta_i \in S} \Delta_i$. Let O be the strict total order over S for which $\Delta_i < \Delta_j$ for $i < j$. Let also $H = \{h_1, \ldots, h_k\}$ be a set of k hash functions such that each $h_i \in H : \{0,1\}^* \to \{1, \ldots, m\}$, that is, each hash function in H takes binary strings as input and outputs a random number uniformly chosen in $\{1, \ldots, m\}$. We define the* Spatial Bloom Filter *(SBF) over (S, O) as the set of couples*

$$B^{\#}(S,O) = \bigcup_{i \in I} \langle i, \max L_i \rangle \ , \tag{4.9}$$

where I is the set of all values output by hash functions in H for elements of $\bar{S}$

$$I = \bigcup_{\delta \in \bar{S}, h \in H} h(\delta) \ , \tag{4.10}$$

and L_i is the set of labels l such that:

$$L_i = \{l \mid \exists \delta \in \Delta_l, \exists h \in H : h(\delta) = i\} \ . \tag{4.11}$$

A spatial Bloom filter $B^{\#}(S,O)$ can be represented as a vector $b^{\#}$ composed of m values, where the i-th value

$$b^{\#}[i] = \begin{cases} l & \text{if } \langle i, l \rangle \in B^{\#}(S,O) \\ 0 & \text{if } \langle i, l \rangle \notin B^{\#}(S,O) \end{cases} \ . \tag{4.12}$$

In the following, when referring to a SBF, we refer to its vector representation $b^{\#}$.

A SBF is built as follows. Initially all values in $b^{\#}$ are set to 0. Then, for each element $\delta \in \Delta_1$ and for each $h \in H$ we calculate $h(\delta) = i$, and set the i-th value of $b^{\#}$ to 1 (that is, to the label of Δ_1). The same is done for the elements belonging to the set Δ_2, setting $b^{\#}[i]$ to 2. We proceed incrementally until all sets in S have been encoded in $b^{\#}$. We observe that, following Definition 5, should a collision occur, the label with higher value is the one stored at the end of the process. Thus, values in the filter corresponding the elements in Δ_s will never be overwritten. This procedure is formalized in Algorithm 4.2 and depicted in Figure 4.7.

The verification process shall check whether an element δ_u is contained in a set $\Delta_i \in S$. Hence we verify whether $\delta_u \in \Delta_i$ if

$$\exists h \in H : b^{\#}[h(\delta_u)] = i \quad \text{and} \quad \forall h \in H, b^{\#}[h(\delta_u)] \geq i \ . \tag{4.13}$$

The procedure is described in Algorithm 4.3.

In practice, if any value of $b^{\#}$ in a position that corresponds to the output of one of the hash functions for δ_u is 0, then $\delta_u \notin \bar{S}$. If all the hashes map to elements

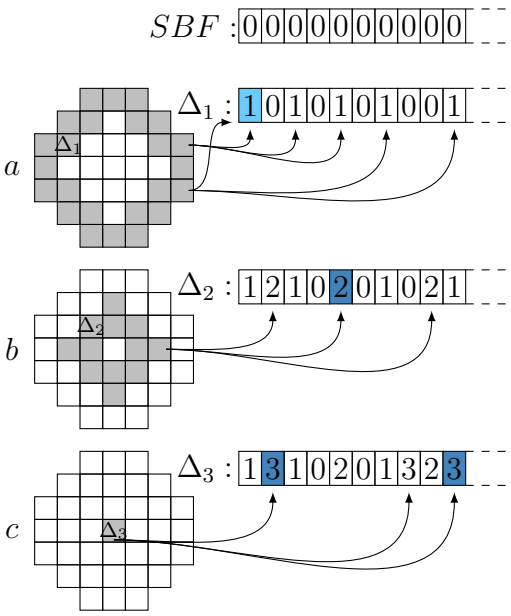


Figure 4.7: Areas Δ_1, Δ_2 and Δ_3 are used to construct a SBF. Three hash functions are used to map each element into the filter. Only the first ten elements of the SBF are shown. In a, two elements belonging to Δ_1 are processed by the hash functions, resulting in six 1 value elements to be written into the SBF. The first element collides as highlighted. This kind of collision is the same that may occur in a classic Bloom Filter. After each element in Δ_1 is processed, the algorithm processes elements in Δ_2 (b) and finally in Δ_3 (c). Note that the collisions in b and c are different from the previous one and are SBF specific. Areas marked with a greater label are assumed to be more important from the provider point of view and overwrite elements of lower value on collision.

of value i, then $\delta_u \in \Delta_i$ minus a false positive probability which is discussed in the following. The same applies if at least one hash maps to an element of value i and the remaining hashes map to elements of value $> i$. In fact, since when a collision occurs the highest value is stored, a lower value could be overwritten.

Similarly to the case of the original Bloom filter (Section 4.3.1), a false positive probability p exists when determining whether an element belongs to the set $\bar{S}$ or not. In the case of a spatial Bloom filter $B^{\#}(S, O)$, however, the probability p can be divided into several probabilities p_i, each one subset-specific. Specifically, p_i is the probability that an element δ is wrongly recognized as belonging to the set Δ_i, while either $\delta \notin \bar{S}$ or $\delta \in \Delta_j$, with $j < i$. For instance, a false positive assigned to the set Δ_s occurs if each hash collides with a value s in $b^{\#}$. Thus we can denote this probability as follows:

$$p_s \approx \left(1 - e^{-\frac{k|\Delta_s|}{m}}\right)^k \quad . \tag{4.14}$$

Algorithm 4.2: Spatial Bloom Filter construction.

Input: $\Delta_1, \Delta_2, \ldots, \Delta_s, H$;
Output: $b^{\#}$;

```
for i ← 1 to s do
    foreach δ ∈ Δ_i do
        foreach h ∈ H do
            b^#[h(δ)] ← i;
        end
    end
end
return b^#;
```

Algorithm 4.3: Spatial Bloom Filter verification.

Input: $b^{\#}, H, \delta_u, s$;
Output: Δ_i;

```
i = s;
foreach h ∈ H do
    if b^#[h(δ_u)] = 0 then
        return false;
    else
        if b^#[h(δ_u)] < i then
            i ← b^#[h(δ_u)];
        end
    end
end
return Δ_i;
```

Similarly, we can compute the probability to wrongly assign an element to the set Δ_{s-1} considering all of the possible collisions with elements belonging to Δ_s and Δ_{s-1}, excluding those deriving from collisions with elements belonging to Δ_s entirely. Hence

$$p_{s-1} \approx \left(1 - e^{-\frac{k|\Delta_s \cup \Delta_{s-1}|}{m}}\right)^k - p_s \ . \tag{4.15}$$

We can proceed likewise to the last set:

$$p_1 \approx \left(1 - e^{-\frac{k|\bar{S}|}{m}}\right)^k - p_s - p_{s-1} - \cdots - p_2 \ . \tag{4.16}$$

It follows that $p_1 + p_2 + \cdots + p_s = p$, where p is the same false positive probability provided in (4.4) if $|\bar{S}| = n$.

In the following it is assumed that the possibility of false positives among sets (that is, having elements in $\bar{S}$ assigned to the wrong set) is deemed as generally acceptable when using a SBF.

Finally, note that a SBF bears some resemblance to a bloomier filter [27], a variant of the classical Bloom filter used for storing binary functions instead of sets. The originating sets could in fact be defined through a function, and the corresponding bloomier filter could be built accordingly. However, in the case of a spatial Bloom filter we have an error probability between different Δ's, but we know exactly whether a $\delta \in S$ or not. A bloomier filter, instead, would behave in the opposite way: the function always outputs the correct Δ, but there exists a probability that a $\delta \notin S$ will be wrongly recognized as belonging to one Δ. Considering location-aware applications, it is reasonable to assume an error in positioning over two contiguous

areas of interest as acceptable, while mistakenly recognizing a position outside the areas of interests (even by far) as inside as much more problematic. Therefore, the author believes that the proposed spatial Bloom filters are better suited to be used in the location-aware context, while bloomier filters might still be useful in specific application scenarios.

4.3.5 Private positioning protocols for SBFs

A major feature of SBFs is that they allow private computation of location based information. This is showed by providing two protocols based on spatial Bloom filters that address the problem of location privacy in a location-aware application. In general, a location-aware application is any service that is based on (partial) knowledge of the geographic position of the user. Here, however, the author focuses on applications in which the service provider has an interest in learning when the user is within an area (or close to a point) of interest.

The presented protocols are designed for a secure multi-party computation setting, where the user and the service provider are mutually distrusting, and therefore do not want to disclose private information to the other party. In the case of the user, private information is his exact location. The service provider, instead, does not want to disclose the monitored areas. This problem is addressed by providing a scheme that allows the provider of a service to detect when the user is within an area of interest, without requiring the user to reveal his exact position to the provider. At the same time, the privacy of the provider is also guaranteed with respect to the areas of interest. The privacy benefits for the user are double: first and foremost, the relative location is only revealed when the user is within predetermined areas, and remains private otherwise. Secondly, even when presence in an area is detected, only this generic information is learned by the provider, and not the actual position. Following the area coverage mechanism proposed in Section 4.3.3.1, for instance, the provider learns the distance from the central area to a certain extent, while the direction from which the user approaches it stays private. Dividing the area around the point of interest in a different manner may reveal instead the direction but conceal the distance within the area range.

In the following two different settings are discussed: in the first setting the user communicates directly to the service provider, who computed beforehand a spatial Bloom filter relative to the areas he is interested in monitoring. In the second setting, instead, the service provider computes the BF, but communication with the user is handled by a third party, to which the provider outsources the task. In both setting,

no trust is implied among the parties, including the third party, and it is assumed the parties do not collude with each other. The honest-but-curious setting is assumed, as defined in the preliminaries (Section 4.3.2).

4.3.5.1 Two-party protocol

In the two-party scenario the communication happens between the service provider *Paul* and the user *Ursula*. It is assumed the user has access to a positioning system that allows her to determine her geographic position. Ursula is interested in using a location-aware service provided by Paul, but she does not want to disclose her exact position. Paul, on the other hand, wants to learn if Ursula is close to some points of interest or is within an area of interest, but he does not want to share with her these locations. Since the two parties are mutually distrusting, this is a secure multi-party computation problem.

Protocol 4.1 addresses this problem securely by disclosing only the identifier i of the area Δ_i in which the user is. Intuitively, the protocol works as follows. Paul creates a SBF for the points and areas of interest as described in the previous sections. He encrypts the filter (by encrypting each value therein) with an encryption scheme that allows the private Hadamard product defined in Algorithm 4.1, and sends it to Ursula. Ursula creates a SBF for the set composed only of her position in the grid. The filter is binary, since 0's and 1's are the only possible values in a filter with only one point of interest. Then Ursula computes the entrywise homomorphic product of the received SBF with one she just computed, she shuffles the values in the resulting encrypted filter and sends the randomly ordered filter back to Paul.

Security Definition In a two-party setting implementing Protocol 4.1, the computation is achieved privately if at the end of the protocol execution Paul learns only $i \in \{1, \ldots, s\}$, and Ursula learns nothing.

Security Analysis As stated in the security definition, a successful execution of Protocol 4.1 should guarantee three conditions: correctness of the result for Paul, privacy for Ursula's position and privacy of the areas encoded in the filter by Paul. These three conditions are discussed in the following.

The protocol ends correctly if the number of non-zero values read in the decrypted $e^{\#}$ by Paul is $< z$ in case Ursula is outside the areas of interests; in case Ursula is within an area, the protocol ends correctly if the number of non-zero values is equal to z, and the area is identified by the smallest non-zero value, minus error probability

Protocol 4.1: Two-party private positioning protocol between service provider Paul and user Ursula.

Before any communication, the provider selects the areas of interest $\Delta_1, \ldots, \Delta_s \subset \mathcal{E}$. Then, he selects the desired false positive probability p, and determines k and m according to (4.5) and (4.6) respectively. Finally, following the notation of Definition 5, the provider computes the spatial Bloom filter $b^{\#}$ over $\bar{S}$ using Algorithm 4.2.

1. The service provider Paul generates a public and private key pair using a multiplicative homomorphic encryption scheme, and sends the public key to the user Ursula.
2. Paul sends to Ursula the encryption of the precomputed SBF $\text{Enc}\left(b^{\#}\right)$, the set of k hash functions H, the value m and the conventional grid $\mathcal{E}$.
3. At regular time intervals, or when required by the specific application, Ursula determines her geographic position and selects the corresponding grid region $e_u \in \mathcal{E}$. Then, following Algorithm 4.2 and using the values and functions shared by Paul, she builds a spatial Bloom filter $b_u^{\#}$ over $\{e_u\}$ and counts the number z of values equals to 1 therein.
4. Ursula computes $e^{\#} = \text{Enc}\left(b^{\#}\right) \boxdot b_u^{\#}$ using the homomorphic properties of the encryption scheme (Algorithm 4.1). Then she applies a random permutation to the values in the filter, and sends z and the result to Paul.
5. Paul decrypts $e^{\#}$ and counts all non-zero values. If the resulting number is $< z$, Ursula's position is outside of the areas on which the SBF was built. Otherwise, the value i, corresponding to area Δ_i identifying Ursula's position (minus error probability p_i), is the smallest non-zero value in $\text{Dec}\left(e^{\#}\right)$.

p_i. The former case is always true, for the properties of Definition 5, as explained in Section 4.3.4. In the latter case, the false positive probability p_i for each area i is determined by Paul according to (4.16) during filter creation. It is therefore Paul himself who decides the correctness bounds of the protocol.

The second condition (Ursula's privacy) is respected if Paul learns only in which (predefined) area the user is, and not her exact position at the end of the protocol. If the user is outside the areas of interest, the provider should learn nothing. Ursula encodes her position in $b_u^{\#}$ at step 3 of the protocol, and sends the encrypted filter $e^{\#} = \text{Enc}\left(b^{\#}\right) \boxdot b_u^{\#}$ back to Paul after performing a random permutation on the order of its values. The homomorphic properties of a public key encryption scheme guarantee that Paul can only learn a number of values from $b^{\#}$ that corresponds to non-zero values in $b_u^{\#}$ [60]. At the same time, the random permutation prevents him from understanding to which position in $b^{\#}$ each of these values corresponds to, therefore making it impossible to reconstruct Ursula's filter based on the order of

elements. If the number of non-zero values is z, and all take the value i corresponding to an area of interest, Paul only learns the area of interest. In case, instead, some values are $> i$ for some of the positions on the grid within the area of interest, then Paul learns the area of interest Δ_i and a pattern of values. The same applies in case Ursula is outside of any area of interest, but the decryption of $e^{\#}$ reveals a number of non-zero values $w < z$. In the following the focus is posed on the latter scenario, as a potential attack exploiting the pattern information could reveal the user's position even when she is outside the areas of interests. In fact, if the pattern is unique for a position on the grid, Paul may be able to learn Ursula's position by performing an exhaustive search on all the possible positions on the grid: given the irreversibility of (spatial) Bloom filters, the complexity of the attack is linear to the number of such positions. We prevent this attack by having each pattern shared by at least a possible positions: in which case we achieve a-anonimity for the user's position even in case of an exhaustive search. We define an arbitrarily small security parameter ϵ, and we consider the privacy condition to be met if the probability of Paul learning Ursula's position is $\frac{1}{a} < \epsilon$. For each number $w \in \{1, \ldots, z\}$ of non-zero values obtained by Paul, we can estimate the value of a based on the number of possible positions in $\mathcal{E}$ and the number of areas of interest s. In particular, the number of possible patterns for a given w is computed as the combinations with repetitions of length w, $\binom{s+w-1}{w}$. Based on this, it is possible to estimate the average value $\bar{a}$ for the different a's of all possible combination with repetitions to be

$$\bar{a} = \frac{|\mathcal{E}|}{\sum_{w=1}^{k} \binom{s+w-1}{w} + 1} \quad , \tag{4.17}$$

if we assume a linear distribution of the values $\{1, \ldots, s\}$ over the filter. The security condition is hence met if $\frac{1}{a} < \epsilon$ for all a's relative to any possible w. We note, from the formula above, that this mostly depends on the number of areas of interest s and, on a lesser extent, on the number of hashes k (since $z \leq k$). These two values can therefore be tuned in order to achieve the desired security parameter ϵ, as both values are selected before the creation of the filter. Considering the order of magnitude of $|\mathcal{E}|$, which is 10^{12}, an appropriately built filter can satisfy a security parameter $\epsilon = 10^{-6}$ for most values of k and s. Thanks to the fine grained nature of the grid, even geographically limited settings which restricts the area of potential positions of the user can achieve reasonable security margins ($\epsilon \approx 10^{-3}$): in fact, small areas of a few square kilometers already include several millions possible positions (Section 4.3.3).

	User	**Provider**	**Third party**
Comp. (2-p)	1 SBF-insertion, 1 Private Hadamard Product	1 decryption, 1 match count	
Comp. (3-p)	k hashes	1 decryption, 1 match count	1 SBF-completion, 1 Private Hadamard Product
Comm. (2-p)	$\mathcal{O}(m)$ $(\lfloor \log_2 s \rfloor + 1)\, m$	$\mathcal{O}(m)$ $(\lfloor \log_2 s \rfloor + 1)\, m$	
Comm. (3-p)	$\mathcal{O}(\log_2 m)$ $k\,(\lfloor \log_2 m \rfloor + 1)$	$\mathcal{O}(m)$ $(\lfloor \log_2 s \rfloor + 1)\, m$	$\mathcal{O}(m)$ $k\,(\lfloor \log_2 m \rfloor + 1) + (\lfloor \log_2 s \rfloor + 1)\, m$

Table 4.6: Computation and communication load for stakeholders.

Finally, the privacy of the service provider, that is, the secrecy of the areas encoded in the filter, is ensured by the encryption of the filter itself. Ursula, in fact, never learns the cleartext of the filter, as she is able to perform the multiplication of step 4 in the encrypted domain thanks to the homomorphic properties of the public key encryption scheme.

Computation and Communication Analysis The computational complexity for the insertion and the verification of a single element in a SBF are linear in the number k of hash functions used for the filter. The private Hadamard product has instead a computational cost linear to the length of the filter m.

Since this primitive is intended to be used in concrete scenarios, in the following an evaluation of actual communication costs, and number of computational operations to be performed during the execution of the protocol are provided (Table 4.6). While being a generally compact data structure, a SBF built over a significantly large number of sets can consume a sizeable amount of memory. While m bits are required for storing a classical Bloom filter b, a SBF needs more bits due to the labels relative to the subsets Δ_i. More precisely, in order to store $b^{\#}$, $(\lfloor \log_2 s \rfloor + 1)\, m$ bits are needed. Depending on the number of areas and the desired error probability, a SBF could require a storage space (and communication cost when transmitted) not suitable for constrained scenarios, as in the case of mobile devices. For instance, consider hash functions with a 16 bit digest (i.e. $m = 2^{16}$) and an area of interest divided into six sub areas. Since $s = 6$, a SBF built on these functions needs $(\lfloor \log_2 6 \rfloor + 1)\, 2^{16}$ bits, resulting in approximately 24 KB data structure. For this reason a protocol involving a third party which offloads user's bandwidth consumption is introduced in the next section.

Protocol 4.2: Three-party private positioning protocol among provider Paul, third party Olga and user Ursula.

Before any communication, the provider selects the areas of interest and creates the corresponding spatial Bloom filter similarly to Protocol 4.1.

1 The service provider Paul generates a public and private key pair using a multiplicative homomorphic encryption scheme, and sends the public key to the third party Olga.
2 Paul sends to Olga the encryption of the precomputed spatial Bloom filter $\mathrm{Enc}\left(b^{\#}\right)$ and the value m. Then, Paul sends to the user Ursula the set of k hash functions H and the conventional grid $\mathcal{E}$.
3 At regular time intervals, or when required by the specific application, Ursula determines her geographic position and selects the corresponding grid region $e_u \in \mathcal{E}$. Then, she computes the values $\{v_1, \ldots, v_k\}$ where $v_i = h_i(e_u)$, and sends them to Olga.
4 Olga receives the values from Ursula and builds $b_o^{\#}$, by assigning $b_o^{\#}[v_i] = 1$ for every $v_i \in \{v_1, \ldots, v_k\}$. Then, she calculates z as the number of 1's in $b_o^{\#}$.
5 Olga computes $e^{\#} = \mathrm{Enc}\left(b^{\#}\right) \boxdot b_o^{\#}$ using the homomorphic properties of the encryption scheme (Algorithm 4.1). Then she applies a random permutation to the values in the filter, and sends z and the result to Paul.
6 Paul decrypts $e^{\#}$ and counts all non-zero values. If the resulting number is $< z$, Ursula's position is outside of the areas on which the SBF was built. Otherwise, the value i, corresponding to Ursula's area Δ_i (minus error probability p_i), is the smallest non-zero value in $\mathrm{Dec}\left(e^{\#}\right)$.

4.3.5.2 Three-party protocol

In the three-party scenario the communication does not happen directly between the service provider and the user. The service provider is responsible for creating and managing the filter, but the verification of user values and therefore all direct communication with the user is outsourced to a third party, here called *Olga*. The third party is introduced in order to decrease the computation and communication burden imposed on the user Ursula. In fact, while it is reasonable to assume that the service provider has adequate resources in terms of computational power and bandwidth to manage filters of big size, the same assumption can not be made for the user, who might be constrained to the limited resources of a mobile device such as a smartphone. Therefore, all onerous tasks are offloaded to the provider and the third party, who is also assumed to be communication and computationally capable.

Security Definition In a three-party setting implementing Protocol 4.2, assuming that no information other than the one implied by the protocol is shared between the parties (parties do not collude), the computation is achieved privately if at the end of the protocol execution Paul learns only $i \in \{0, \ldots, s\}$, while Olga and Ursula learn nothing.

Security Analysis The security of the three-party protocol follows that of the two-party protocol above. The introduction of the third party means however that the user sends her unencoded hash values to the third party, who performs the private Hadamard product. This exposes the user to an attack on the spatial Bloom filter by the third party. While Bloom filters have proved to be irreversible, an exhaustive search may reveal to Olga the input used to produce the received hash outputs. This attack, however, assumes knowledge of $\mathcal{E}$ by Olga. The conventional grid $\mathcal{E}$ represents in fact the coding scheme (or ordering) of the elements on the geographical grid: that is, which value is to be given as input to the hash functions for each position. Since this information is not required by Olga for the execution of the protocol, the user and the provider can agree on an encoding scheme (which can simply be a random ordering of the geographical grid elements) unknown to the third party, thus preventing her from running a search attack. The same goal can also be achieved by using keyed hash functions, which would however require a key exchange between the two parties.

A second threat to which the user is exposed is due to the deterministic nature of the hash results for the same input. In fact, the third party may easily know if the user is revisiting the same grid position twice by comparing the hash digests. In settings in which this is considered unacceptable, a temporal-based variation of the above encoding of the geographical grid can be used.

Chapter 5

Conclusion

Cities are growing day by day. Along with their dimension and population, each process concerning the urban context involved is getting more complex. Today we need to design infrastructures capable of handling heterogeneous sensor networks in order to integrate the raw data coming from urban area and process them to expose a range of services to the citizen, enterprise, government, and machine to machine (M2M). Public administrations are nowadays engaged in an ensemble of processes aimed at facing this challenge.

An integrated ICT system designed for smarter cities should be able to manage several sensor networks in order to provide citizens and the public administration with a direct interface to wide variety of city aspects. Such systems should include citizen-driven information as well, as that of citizen interaction is a central smart city topic. Citizens are transformed from passive subjects into live actors. Public administration would also experience several benefits due to the combination of sensor driven information and social/financial ones. Combining (or simply view simultaneously) environmental data with tax information, births, deaths, cadastral information and more in general with other types of indicators frequently managed by public administration would enhance management control over the whole city.

Thus, ICT is a key-component in city innovation as it acts as enabling technology for almost all the involved processes. Unfortunately, as discussed in this thesis, this integration does not come with no concerns. The design and implementation of an integrated infrastructure able to combine environmental, social and sensor-driven data in order to enable a real-time management control of the city itself represents an impressive issue and does not come with a fixed installation procedure. Such a model should in fact be customized in order to fit the needs of each specific city.

The studies and experiments carried out and presented in this thesis suggest that, in the near future, industry, institutions and research communities should investigate

in three main directions. The first and most obvious subject relates the improvement of single smart services deployed within the urban context. It may consist for instance in the design of sensors with enhanced capabilities or in the proper design of a web application. A specific focus should be posed to the APIs designed to query the service. In fact, a smart environment often relies on the aggregation of a number of services which need to interact one with each other or simply to be connected to a central system. Thus, to design simple and interoperable APIs is sometimes important as much as to design all of the service features. The second research direction concerns service integration. Often, the real added value in a smart city context comes from the connection of several services already deployed within the city itself. Here the challenge resides in designing operating systems for the city, in the choice and set up of a good architecture upon which run them (for instance in the cloud) and in the definition of procedures able to analyze these collection of heterogeneous data in order to determine summary indicators. Finally, ICT infrastructures along with the exponential diffusion of mobile devices are raising a serious threat concerning people's privacy. Urban context are even more conceived as Living Labs where people provide and consume data to and from the city information system or a part of it. In this citizen-as-a-sensor scenario, the more interesting and powerful feature is often represented by location-aware information, which adds a new spatial dimension to be used in a wide number of contexts. Thus, the study of how to preserve user's and provider's privacy without limiting the services involved should be faced as one of the near future biggest challenges. In other words, we should protect the privacy of tomorrow's adults, i.e. those young people of today used to disclose an impressive amount of personal information through social networks, web applications and several other kinds of smart services.

Appendix A

List of scientific publications

During his doctoral program, the author contributed to the publications listed below:

- Luca Calderoni, Matteo Ferrara, Annalisa Franco, and Dario Maio. Indoor localization in a hospital environment using random forest classifiers. *Expert Syst. Appl.*, 42(1):125–134, 2015
- Paolo Palmieri, Luca Calderoni, and Dario Maio. Spatial bloom filters: Enabling privacy in location-aware applications. In Dongdai Lin, Moti Yung, and Jianying Zhou, editors, *Information Security and Cryptology - 10th International Conference, Inscrypt 2014, Beijing, China, December 13-15, 2014, Revised Selected Papers*, Lecture Notes in Computer Science. Springer, 2014
- S. Gaitan, L. Calderoni, P. Palmieri, M.t. Veldhuis, D. Maio, and M.B. van Riemsdijk. From sensing to action: Quick and reliable access to information in cities vulnerable to heavy rain. *Sensors Journal, IEEE*, 14(12):4175–4184, Dec 2014
- Gildas Avoine, Luca Calderoni, Jonathan Delvaux, Dario Maio, and Paolo Palmieri. Passengers information in public transport and privacy: Can anonymous tickets prevent tracking? *International Journal of Information Management*, 34(5):682–688, 2014
- Luca Calderoni and Dario Maio. Cloning and tampering threats in e-passports. *Expert Syst. Appl.*, 41(11):5066–5070, 2014
- Luca Calderoni, Dario Maio, and Stefano Rovis. Deploying a network of smart cameras for traffic monitoring on a "city kernel". *Expert Syst. Appl.*, 41(2):502–507, 2014
- Luca Calderoni, Dario Maio, and Paolo Palmieri. Location-aware mobile services for a smart city: Design, implementation and deployment. *JTAER*, 7(3), 2012

Bibliography

[1] KNMI product catalogus, 2013.

[2] 112Meldingen.nl. 112MELDINGEN.NL | 112 meldingen & p2000 alarmeringen van de hulpdiensten online volgen., 2014.

[3] Daniel J Abadi. Data management in the cloud: Limitations and opportunities. *IEEE Data Eng. Bull.*, 32(1):3–12, 2009.

[4] M. Al-Hader, A. Rodzi, A.R. Sharif, and N. Ahmad. Smart city components architicture. In *Computational Intelligence, Modelling and Simulation, 2009. CSSim '09. International Conference on*, pages 93 –97, sept. 2009.

[5] Mahdi Asadpour and Mohammad Torabi Dashti. A privacy-friendly rfid protocol using reusable anonymous tickets. In *10th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-11)*, pages 206–213, Changsha, China, 2011. IEEE Computer Society.

[6] R.M. Ashley, D.J. Balmfort, A.J. Saul, and J.D. Blanskby. Flooding in the future - predicting climate change, risks and responses in urban areas. *Water Science and Technology*, 52(5):265–273, 2005.

[7] Gildas Avoine, Luca Calderoni, Jonathan Delvaux, Dario Maio, and Paolo Palmieri. Passengers information in public transport and privacy: Can anonymous tickets prevent tracking? *International Journal of Information Management*, 34(5):682–688, 2014.

[8] Gildas Avoine, Tania Martin, and Jean-Pierre Szikora. Lire son passe navigo en un clin d'œil. *Multi-System & Internet Security Cookbook – MISC*, 48, March–April 2010.

[9] Michael Barbaro and Tom Jr. Zeller. A face is exposed for aol searcher no. 4417749. *The New York Times*, 9 August 2006. Avbl.: `http://www.nytimes.com/2006/08/09/technology/09aol.html`, 2006.

[10] Karolina Berggren, Mats Olofsson, Maria Viklander, Gilbert Svensson, and Anna-Maria Gustafsson. Hydraulic impacts on urban drainage systems due to changes in rainfall caused by climatic change. *Journal of Hydrologic Engineering*, 17(1):92–98, 2012.

[11] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.

[12] Jeffrey R. Blum, Daniel G. Greencorn, and Jeremy R. Cooperstock. Smartphone sensor reliability for augmented reality applications. In *MobiQuitous*, volume 120 of *LNICST*, pages 127–138. Springer, 2012.

[13] Andrew J. Blumberg and Peter Eckersly. On Locational Privacy, and How to Avoid Losing it Forever. https://www.eff.org/wp/locational-privacy, April 2009.

[14] Vinayak R. Borkar, Michael J. Carey, and Chen Li. Big data platforms: what's next? *ACM Crossroads*, 19(1):44–49, 2012.

[15] P. Borokhov, S. Blandin, S. Samaranayake, O. Goldschmidt, and A. Bayen. An adaptive routing system for location-aware mobile devices on the road network. In *14th International IEEE Conference on Intelligent Transportation Systems (ITSC 2011)*, pages 1839 –1845, October 2011.

[16] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[17] A. T. H Bruin and Koninklijk Nederlands Meteorologisch Instituut. *Veranderingen in neerslagkarakteristieken in Nederland gedurende de periode 1901-2001.* Koninklijk Nederlands Meteorologisch Instituut, De Bilt, 2002.

[18] Bundesamt für Sicherheit in der Informationstechnik. Advanced Security Mechanisms for Machine Readable Travel Documents. Technical report, BSI, March 2012.

[19] Angela Sara Cacciapuoti, Francesco Calabrese, Marcello Caleffi, Giusy Di Lorenzo, and Luigi Paura. Human-mobility enabled wireless networks for emergency communications during special events. *Pervasive and Mobile Computing*, 9(4):472–483, 2013.

[20] Francesco Calabrese, Kristian Kloeckl, and Carlo Ratti. WikiCity: Real-Time Location-Sensitive Tools for the City. In *Digital Cities 5: Urban Informatics, Locative Media and Mobile Technology in Inner-City Developments*. Workshop, 2007.

[21] Luca Calderoni, Matteo Ferrara, Annalisa Franco, and Dario Maio. Indoor localization in a hospital environment using random forest classifiers. *Expert Syst. Appl.*, 42(1):125–134, 2015.

[22] Luca Calderoni and Dario Maio. Cloning and tampering threats in e-passports. *Expert Syst. Appl.*, 41(11):5066–5070, 2014.

[23] Luca Calderoni, Dario Maio, and Paolo Palmieri. Location-aware mobile services for a smart city: Design, implementation and deployment. *JTAER*, 7(3), 2012.

[24] Luca Calderoni, Dario Maio, and Stefano Rovis. Deploying a network of smart cameras for traffic monitoring on a "city kernel". *Expert Syst. Appl.*, 41(2):502–507, 2014.

[25] Cassa Depositi e Prestiti. Smart City: Progetti di sviluppo e strumenti di finanziamento. Technical report, September 2013.

[26] Manville Catriona, Cochrane Gavin, Cave Jonathan, Millard Jeremy, Pederson Jimmy Kevin, Thaarup Rasmus Kare, Liebe Andrea, Wissner Matthias, Massink Roel, and Kotterink Bas. Mapping Smart Cities in the EU, January 2014.

[27] Bernard Chazelle, Joe Kilian, Ronitt Rubinfeld, and Ayellet Tal. The bloomier filter: an efficient data structure for static support lookup tables. In *SODA*, pages 30–39. SIAM, 2004.

[28] Douglas Comer. The Ubiquitous B-Tree. *ACM Computing Surveys*, 11(2):121–137, 1979.

[29] Carlo Curino, Evan P. C. Jones, Raluca A. Popa, Nirmesh Malviya, Eugene Wu, Samuel Madden, Hari Balakrishnan, and Nickolai Zeldovich. Relational Cloud: a Database Service for the cloud. In *CIDR*, pages 235–240. www.crdrdb.org, 2011.

[30] Louis de Grange, Felipe González, Juan Carlos Mu noz, and Rodrigo Troncoso. Aggregate estimation of the price elasticity of demand for public transport in integrated fare systems: The case of transantiago. *Transport Policy*, 29(0):178 – 185, 2013.

[31] Yves-Alexandre de Montjoye, César A. Hidalgo, Michel Verleysen, and Vincent D. Blondel. Unique in the Crowd: The privacy bounds of human mobility. *Scientific Reports*, 3, March 2013.

[32] Claudia Diaz and Seda Gurses. Understanding the landscape of privacy technologies. pages 58–63, 2012.

[33] John K. Dixon. Pattern recognition with partly missing data. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(10):617–621, 1979.

[34] John Domingue, Alex Galis, Anastasius Gavras, Theodore Zahariadis, Dave Lambert, Frances Cleary, Petros Daras, Srdjan Krco, Henning Müller, Man-Sze Li, Hans Schaffers, Volkmar Lotz, Federico Alvarez, Burkhard Stiller, Stamatis Karnouskos, Susanna Avessta, and Michael Nilsson, editors. *The Future Internet - Future Internet Assembly 2011: Achievements and Technological Promises*, volume 6656 of *Lecture Notes in Computer Science*. Springer, 2011.

[35] European Commission. Advancing and Applying Living Lab Methodologies. Technical report, 2010.

[36] R.J. Firmino, F. Duarte, and C. Ultramari. *ICTs for Mobile and Ubiquitous Urban Infrastructures: Surveillance, Locative Media, and Global Networks*. Information Science Reference, 2011.

[37] S. Gaitan, L. Calderoni, P. Palmieri, M.t. Veldhuis, D. Maio, and M.B. van Riemsdijk. From sensing to action: Quick and reliable access to information in cities vulnerable to heavy rain. *Sensors Journal, IEEE*, 14(12):4175–4184, Dec 2014.

[38] Megan Geuss. Japanese railway company plans to sell data from e-ticket records. *Ars Technica*, 7 July 2013. Avbl.: `http://arstechnica.com/business/2013/07/japanese-railway-company-plans-to-sell-data-from-e-ticket-records/`, 2013.

[39] Rudolf Giffinger, Christian Fertner, Hans Kramar, Robert Kalasek, Nataša Pichler-Milanović, and Evert Meijers. Smart Cities: Ranking of European Medium-sized Cities, 2007.

[40] Google Inc. Google maps api v3 reference. `http://code.google.com/apis/maps/documentation/v3/reference.html`, 2012.

[41] Yanying Gu, Anthony Lo, and Ignas G. Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys and Tutorials*, 11(1):13–32, 2009.

[42] Antonin Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In Beatrice Yormark, editor, *SIGMOD Conference*, pages 47–57. ACM Press, 1984.

[43] Jane K Hart and Kirk Martinez. Environmental sensor networks: A revolution in the earth system science? *Earth-Science Reviews*, 78:177–191, 2006.

[44] Oliver Haubensak. Smart Cities and Internet of Things. In *Business Aspects of the Internet of Things, Seminar of Advanced Topics*, pages 33–39. ETH Zurich, 2011.

[45] L. Hentila, A. Taparungssanagorn, H. Viittala, and M. Hamalainen. Measurement and modelling of an uwb channel at hospital. In *Ultra-Wideband, 2005. ICU 2005. 2005 IEEE International Conference on*, 2005.

[46] José M. Hernández-Muñoz, Jesús Bernat Vercher, Luis Muñoz, José Antonio Galache, Mirko Presser, Luis A. Hernández Gómez, and Jan Pettersson. Smart Cities at the Forefront of the Future Internet. In Domingue et al. [34], pages 447–462.

[47] Thomas S. Heydt-Benjamin, Hee-Jin Chae, Benessa Defend, and Kevin Fu. Privacy for public transportation. In George Danezis and Philippe Golle, editors, *Privacy Enhancing Technologies*, volume 4258 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2006.

[48] Mahsa Honary, Lyudmila Mihaylova, and Costas Xydeas. Practical classification methods for indoor positioning. *The Open Transportation Journal*, 6:31–38, 2012.

[49] Information Technology Laboratory and NIST. *Secure hash standard (SHS)*. U.S. Dept. of Commerce, National Institute of Standards and Technology, Gaithersburg, MD, 2008.

[50] International Civil Aviation Organization. Machine readable travel documents. Technical report, 2006.

[51] International Telecommunication Union. The Internet of Things. ITU Internet Report, 2005.

[52] International Telecommunication Union. Industrial, scientific and medical (ISM) applications (of radio frequency energy). Technical Report, October 2009.

[53] Istituto Nazionale di Statistica (ISTAT). Focus: Trasporti urbani. *Istat*, 5 April 2011. Avbl.: `http://www.ontit.it/opencms/export/sites/default/ont/it/documenti/files/ONT\_2011-04-06\_02603.pdf`, 2011.

[54] ITU-T TSAG. A Preliminary Study on the Ubiquitous Sensor Network. Technical report, 2007.

[55] N. Janssen. Assessing the quality of disdrometer data for measuring rainfall in the rotterdam region and modelling the spatial correlation for short-term rainfall accumulation intervals. Technical report, Wagening University and Reseacch Centre, Netherlands, 2013.

[56] Wei Jiang and Chris Clifton. A secure distributed framework for achieving k-anonymity. *VLDB J.*, 15(4):316–333, 2006.

[57] Dai Jiazhu and Li Zhilong. A location authentication scheme based on proximity test of location tags. In *ICINS 2013*, pages 1–6, Nov 2013.

[58] V Kastrinaki, M Zervakis, and K Kalaitzakis. A survey of video processing techniques for traffic applications. *Image and Vision Computing*, 21(4):359 – 381, 2003.

[59] Florian Kerschbaum, Hoon Wei Lim, and Ivan Gudymenko. Privacy-preserving billing for e-ticketing systems in public transportation. In *Workshop on Privacy in the Electronic Society*, 2013.

[60] Hiroaki Kikuchi and Jun Sakuma. Bloom filter bootstrap: Privacy-preserving estimation of the size of an intersection. *JIP*, 22(2):388–400, 2014.

[61] Hyun Hee Kim, Kyoung Nam Ha, Suk Lee, and Kyung Chang Lee. Resident location-recognition algorithm using a bayesian classifier in the pir sensor-based indoor location-aware system. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 39(2):240–245, 2009.

[62] Cornel Klein and Gerald Kaefer. From Smart Homes to Smart Cities: Opportunities and Challenges from an Industrial Perspective. In Sergey Balandin, Dmitri Moltchanov, and Yevgeni Koucheryavy, editors, *NEW2AN*, volume 5174 of *Lecture Notes in Computer Science*, page 260. Springer, 2008.

[63] Lawrencem A. Klein, Milton K. Mills, and David R.P. Gibson. Traffic detector handbook: Third edition. Technical report, Federal Highway Administration, oct 2006.

[64] N. Komninos. *The Age of Intelligent Cities: Smart Environments and Innovation-for-all Strategies*. Regions and Cities. Taylor & Francis, 2014.

[65] Hakan Koyuncu and Shuang Hua Yang. A survey of indoor positioning and object locating systems. *International Journal of Computer Science and Network Security*, 10(5):121–128, 2010.

[66] Andreas Krause, Ram Rajagopal, Anupam Gupta, and Carlos Guestrin. Simultaneous optimization of sensor placements and balanced schedules. *IEEE Trans. Automat. Contr.*, 56(10):2390–2405, 2011.

[67] Lars Kulik. Privacy for real-time location-based services. *SIGSPATIAL Special*, 1(2):9–14, 2009.

[68] Sang-Ho Lee, Tan Yigitcanlar, Jung-Hoon Han, and Youn-Taik Leem. Ubiquitous urban infrastructure: Infrastructure planning and development in korea. *Innovation: Management, Policy & Practice*, 10:282–292, 2008.

[69] Xiaolei Li, Jiawei Han, Jae-Gil Lee, and Hector Gonzalez. Traffic density-based discovery of hot routes in road networks. In Dimitris Papadias, Donghui Zhang, and George Kollios, editors, *SSTD*, volume 4605 of *Lecture Notes in Computer Science*, pages 441–459. Springer, 2007.

[70] M. Litzenberger, B. Kohn, G. Gritsch, N. Donath, C. Posch, N.A. Belbachir, and H. Garn. Vehicle counting with an embedded traffic data system using an optical transient sensor. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 36 –40, 30 2007-oct. 3 2007.

[71] H. Liu, Houshang Darabi, Pat P. Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 37(6):1067–1080, 2007.

[72] Jingwei Liu, Yong Zhao, Yule Yuan, Wei Luo, and Kai Liu. Vehicle capturing and counting using a new edge extraction approach. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 62 –66, oct. 2011.

[73] Siyuan Liu, Yunhuai Liu, Lionel M. Ni, Jianping Fan, and Minglu Li. Towards mobility-based clustering. In Bharat Rao, Balaji Krishnapuram, Andrew Tomkins, and Qiang Yang, editors, *KDD*, pages 919–928. ACM, 2010.

[74] Zhan Liu, Riccardo Bonazzi, Boris Fritscher, and Yves Pigneur. Privacy-friendly business models for location-based mobile services. *JTAER*, 6(2):90–107, 2011.

[75] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[76] Dario Maio and Davide Maltoni. Metodo innovativo per la rilevazione delle ombre prodotte da veicoli in transito, september 2012.

[77] Mauro Martino, Francesco Calabrese, Giusy Di Lorenzo, Clio Andris, Liu Liang, and Carlo Ratti. Ocean of information: fusing aggregate & individual dynamics for metropolitan analysis. In Charles Rich, Qiang Yang, Marc Cavazza, and Michelle X. Zhou, editors, *IUI*, pages 357–360. ACM, 2010.

[78] Jan Philip Matuscheck. Finding Points Within a Distance of a Latitude/Longitude Using Bounding Coordinates. Technical report, 2011.

[79] Rainer Mautz. Indoor positioning technologies. ETH Zürich Habilitation Thesis, 2012.

[80] Ettore Merlo, Dominic Letarte, and Giuliano Antoniol. Automated Protection of PHP Applications Against SQL-injection Attacks. In René L. Krikhaar, Chris Verhoef, and Giuseppe A. Di Lucca, editors, *CSMR*, pages 191–202. IEEE Computer Society, 2007.

[81] Nathalie Mitton, Symeon Papavassiliou, Antonio Puliafito, and Kishor S. Trivedi. Combining cloud and sensors in a smart city environment. *EURASIP J. Wireless Comm. and Networking*, 2012:247, 2012.

[82] Anirban Mondal, Anthony K. H. Tung, and Masaru Kitsuregawa. kNR-tree: a novel R-tree-based index for facilitating spatial window queries on any k relations among N spatial relations in mobile environments. In Panos K. Chrysanthis and George Samaras, editors, *Mobile Data Management*, pages 173–177. ACM, 2005.

[83] Jean Monnerat, Serge Vaudenay, and Martin Vuagnoux. About machine-readable travel documents. In *In Proceedings of the International Conference on RFID Security 2007*, pages 15–28, 2007.

[84] Frank Munz. *Middleware and Cloud Computing: Oracle Fusion Middleware on Amazon Web Services and Rackspace Cloud*. Munz & More Publishing, 2011.

[85] N. Nabian and P. Robinson. *SENSEable City Guide*. SA+P Press, 2011.

[86] Taewoo Nam and Theresa A. Pardo. Conceptualizing smart city with dimensions of technology, people, and institutions. In John Carlo Bertot, Karine Nahon, Soon Ae Chun, Luis F. Luna-Reyes, and Vijay Atluri, editors, *DG.O*, ACM International Conference Proceeding Series, pages 282–291. Digital Government Research Center, 2011.

[87] Taewoo Nam and Theresa A. Pardo. Smart city as urban innovation: focusing on management, policy, and context. In Elsa Estevez and Marijn Janssen, editors, *ICEGOV*, pages 185–194. ACM, 2011.

[88] Milind R. Naphade, Guruduth Banavar, Colin Harrison, Jurij Paraszczak, and Robert Morris. Smarter cities and their innovation challenges. *IEEE Computer*, 44(6):32–39, 2011.

[89] Arvind Narayanan, Narendran Thiagarajan, Mugdha Lakhani, Michael Hamburg, and Dan Boneh. Location privacy via private proximity testing. In *NDSS*. The Internet Society, 2011.

[90] Janus Dam Nielsen, Jakob Illeborg Pagter, and Michael Bladt Stausholm. Location privacy via actively secure private proximity testing. In *PerCom Workshops*, pages 381–386. IEEE, 2012.

[91] Oracle Corporation. Guide to Scaling Web Databases with MySQL Cluster. Technical report, 2012.

[92] M. Oteşteanu, A. Gontean, G. Sârbu-Doagă, and R. Sandra. Software environment for the laser precipitation monitor. *WSEAS Transactions on Information Science and Applications*, 4(1):214–219, 2007.

[93] A. Overeem, A. Buishand, I. Holleman, and R. Uiljenhoet. Statistiek van extreme gebiedsneerslag in neder-land. Technical Report TR332, KNMI, 2012.

[94] Aart Overeem, Hidde Leijnse, and Remko Uijlenhoet. Country-wide rainfall maps from cellular communication networks. *Proceedings of the National Academy of Sciences*, 2013.

[95] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999.

[96] Paolo Palmieri, Luca Calderoni, and Dario Maio. Spatial bloom filters: Enabling privacy in location-aware applications. In Dongdai Lin, Moti Yung, and Jianying Zhou, editors, *Information Security and Cryptology - 10th International Conference, Inscrypt 2014, Beijing, China, December 13-15, 2014, Revised Selected Papers*, Lecture Notes in Computer Science. Springer, 2014.

[97] Xiao Pan and Xiaofeng Meng. Preserving location privacy without exact locations in mobile services. *Frontiers of Computer Science*, 7(3):317–340, 2013.

[98] Jong-Won Park, Chang Ho Yun, Seong Woo Rho, Yongwoo Lee, and Hae-Sun Jung. Mobile Cloud Web-Service for U-City. In *DASC*, pages 1061–1065. IEEE, 2011.

[99] Kun Peng and Feng Bao. A secure rfid ticket system for public transport. In *DBSec*, pages 350–357, 2010.

[100] Paolo Pivato. *Analysis and characterization of Wireless Positioning Techniques in Indoor Environment*. PhD thesis, University of Trento, 2012.

[101] Joan Raventos. System and method for enabling transaction-based service utilizing non-transactional resources, 2001.

[102] R. Reinoso-Rondinel, G. Bruni, and J. A. E. ten Veldhuis. Toward the optimal resolution of rainfall estimates to obtain reliable urban hydrological response: X-band polarimetric radar estimates applied to rotterdam urban drainage system. In *11th International Precipitation Conference 2013*, Ede-Wageningen, 2013. KNMI and WUR.

[103] John Rhoton and Risto Haukioja. *Cloud Computing Architected*. Recursive, Limited, 2011.

[104] Francisca M. Rojas, Kristian Kloeckl, and Carlo Ratti. Dynamic City: Investigations into the sensing, analysis, and applications of real-time, location-based data. In *Proceedings of Harvard GSD Critical Digital Conference*, 2008.

[105] Ahmad-Reza Sadeghi, Ivan Visconti, and Christian Wachsmann. User privacy in transport systems based on rfid e-tickets. In Claudio Bettini, Sushil Jajodia, Pierangela Samarati, and Xiaoyang Sean Wang, editors, *PiLBA*, volume 397 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[106] G. Sagl, T. Blaschke, E. Beinat, and B. Resch. Ubiquitous geo-sensing for context-aware analysis: Exploring relationships between environmental and human dynamics. *Sensors (Switzerland)*, 12(7):9800–9822, 2012.

[107] Gökay Saldamli, Richard Chow, Hongxia Jin, and Bart P. Knijnenburg. Private proximity testing with an untrusted server. In *WISEC*, pages 113–118. ACM, 2013.

[108] Jarno Salonen. Evaluating the security and privacy of near field communication - case: Public transportation. In Souhwan Jung and Moti Yung, editors, *WISA*, volume 7115 of *Lecture Notes in Computer Science*, pages 242–255. Springer, 2011.

[109] Breno Ramos Sampaio, Oswaldo Lima Neto, and Yony Sampaio. Efficiency analysis of public transport systems: Lessons for institutional planning. *Transportation Research Part A: Policy and Practice*, 42(3):445 – 454, 2008.

[110] Corné van der Sande, Sylvie Soudarissanane, and Kourosh Khoshelham. Assessment of relative accuracy of AHN-2 laser scanning data using planar features. *Sensors*, 10(9):8198–8214, 2010.

[111] Hans Schaffers, Nicos Komninos, and Marc Pallot. Smart Cities as Innovation Ecosystems Sustained by the Future Internet. FIREBALL White Paper, April 2012.

[112] Hans Schaffers, Nicos Komninos, Marc Pallot, Brigitte Trousse, Michael Nilsson, and Alvaro Oliveira. Smart Cities and the Future Internet: Towards Cooperation Frameworks for Open Innovation. In Domingue et al. [34], pages 431–446.

[113] J. M. Schuurmans, M. F. P. Bierkens, E. J. Pebesma, and R. Uijlenhoet. Automatic prediction of high-resolution daily rainfall fields for multiple extents: The potential of operational radar. *Journal of Hydrometeorology*, 8(6):1204–1224, 2007.

[114] Kathy S. Schwaig, Albert H. Segars, Varun Grover, and Kirk D. Fiedler. A model of consumers' perceptions of the invasion of information privacy. *Information & Management*, 50(1):1–12, 2013.

[115] F. Seco, A.R. Jimenez, C. Prieto, J. Roa, and K. Koutsou. A survey of mathematical methods for indoor localization. In *Intelligent Signal Processing, 2009. WISP 2009. IEEE International Symposium on*, pages 9–14, 2009.

[116] Jikang Shin and Dongsoo Han. Multi-classifier for wlan fingerprint-based positioning system. In S. I. Ao, Len Gelman, David WL Hukins, Andrew Hunter, and A. M. Korsunsky, editors, *World Congress on Engineering*, pages 768–773. International Association of Engineers, Newswood Limited, 2010.

[117] Jikang Shin, Suk Hoon Jung, Giwan Yoon, and Dongsoo Han. A multi-classifier approach for wifi-based positioning system. In Sio-Iong Ao and Len Gelman, editors, *Electrical Engineering and Applied Computing*, pages 135–147. Springer Netherlands, 2011.

[118] K. Stone, R. Duinen, W. van Veerbeek, and S. Dopp. Sensitivity and vulnerability of urban systems - assessment of climate change impact to urban systems. Technical Report 1202270-008-BGS-0004, Deltares, Netherlands, 2011.

[119] Jingchao Sun, Rui Zhang, and Yanchao Zhang. Privacy-preserving spatiotemporal matching. In *INFOCOM*, pages 800–808. IEEE, 2013.

[120] J.A.E. ten Veldhuis. How the choice of flood damage metrics influences urban flood risk assessment. *Journal of Flood Risk Management*, 4(4):281–287, 2011.

[121] M.C. ten Veldhuis and Birna van Riemsdijk. High resolution weather data for urban hydrological modelling and impact assessment, ict requirements and future challenges. Geophysical Research Abstracts, 2013.

[122] The Nielsen Company. Two Thirds of New Mobile Buyers Now Opting For Smartphones. Technical report, 2012.

[123] Rafael Tonicelli, Bernardo Machado David, and Vinícius de Morais Alves. Universally composable private proximity testing. In *ProvSec*, volume 6980 of *LNCS*, pages 222–239. Springer, 2011.

[124] G. Tseytin, M. Hofmann, D. Lyons, and M. O'Mahony. Tracing individual public transport customers from an anonymous transaction database. *J. of Public Transportation*, 9(4):47–60, 2006.

[125] B. van den Hurk, A. Klein Tank, G. Lenderink, A. van Ulden, G. J. Van Oldenborgh, C. Katsman, H. Van den Brink, F. Keller, J. Bessembinder, G. Burgers, et al. *KNMI climate change scenarios 2006 for the Netherlands*. KNMI De Bilt, 2006.

[126] P. van der Gracht and R. Donaldson. Communication using pseudonoise modulation on electric power distribution circuits. *Communications, IEEE Transactions on*, 33(9):964 – 974, sep 1985.

[127] N. van der Zon. Kwaliteitsdocument AHN-2. Technical Report 1.1, Rijkswaterstaat & Waterschappen, 2011.

[128] E. Viarani. Extraction of traffic information from images at deis. In *Image Analysis and Processing, 1999. Proceedings. International Conference on*, pages 1073 –1076, 1999.

[129] Gabriel Villarubia, Francisco Rubio, Juan F. De Paz, Javier Bajo, and Carolina Zato. Applying classifiers in indoor location system. In Javier Bajo Pérez, Juan M. Corchado Rodríguez, Johannes Fähndrich, Philippe Mathieu, Andrew Campbell, Mari Carmen Suarez-Figueroa, Alfonso Ortega, Emmanuel Adam, Elena Navarro, Ramon Hermoso, and María N. Moreno, editors, *Trends in Practical Applications of Agents and Multiagent Systems*, pages 53–58. Springer International Publishing, 2013.

[130] Stephan von Watzdorf and Florian Michahelles. Accuracy of positioning data on smartphones. In *LocWeb*, page 2. ACM, 2010.

[131] Stephen B. Wicker. The loss of location privacy in the cellular age. *Commun. ACM*, 55(8):60–68, 2012.

[132] Michael Widenius and Davis Axmark. *Mysql Reference Manual*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 1st edition, 2002.

[133] R.L. Wilby. A review of climate change impacts on the built environment. *Built Environment*, 33(1):31–45, 2007.

[134] Jaegeol Yim. Introducing a decision tree-based indoor positioning technique. *Expert Syst. Appl.*, 34(2):1296–1302, 2008.

[135] Sameh Zakhary, Milena Radenkovic, and Abderrahim Benslimane. The quest for location-privacy in opportunistic mobile social networks. In *IWCMC*, pages 667–673. IEEE, 2013.

[136] Daniel Zeng, Christopher C. Yang, Vincent S. Tseng, Chunxiao Xing, Hsinchun Chen, Fei-Yue Wang, and Xiaolong Zheng, editors. *Smart Health - International Conference, ICSH 2013, Beijing, China, August 3-4, 2013. Proceedings*, volume 8040 of *Lecture Notes in Computer Science*. Springer, 2013.

[137] Liping Zhang, S.D. Gupta, Jing-Quan Li, Kun Zhou, and Wei bin Zhang. Path2Go: Context-aware services for mobile real-time multimodal traveler information. In *14th International IEEE Conference on Intelligent Transportation Systems (ITSC 2011)*, pages 174 –179, October 2011.

[138] Yao Zheng, Ming Li, Wenjing Lou, and Y. Thomas Hou. Sharp: Private proximity test and secure handshake with cheat-proof location tags. In *ESORICS*, volume 7459 of *LNCS*, pages 361–378. Springer, 2012.