INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Green IT - People Counting and Detection of Patterns of Movement

António Calçada Novais

Dissertação para a obtenção de Grau de Mestre em

Engenharia de Redes de Comunicações

Júri

Presidente:	Prof. Doutor Paulo Jorge Pires Ferreira
Orientador:	Prof. Doutora Teresa Maria Sá Ferreira Vazão Vasques
Co-orientador:	Prof. Doutor Ricardo Jorge Feliciano Lopes Pereira
Vogais:	Prof. Doutor Artur Miguel do Amaral Arsenio

September 2011

Resumo

O consumo energético hoje em dia desempenha um papel fundamental, não só para a necessidade crescente de redução de custos, mas sobretudo pela esperada redução significativa dos recursos naturais. O conhecimento da localização e dos padrões de mobilidade das pessoas no interior dos edifícios pode ajudar a reduzir o consumo, tornando possível, por exemplo, ajustar o ar condicionado e iluminação para a ocupação real do edifício. Propomos um sistema capaz de contar pessoas e aprender os seus padrões de mobilidade no interior dos edifícios e dar a possibilidade de integrar várias tecnologias. Esta tese também contribui para uma plataforma de monitorização multi-funcional com a integração de um módulo de descoberta de serviços. Neste documento, também analisamos o estado da arte das tecnologias e protocolos utilizados para a descoberta de serviços e localização de pessoas.

Palavras-chave: Contagem de pessoas, Localização de pessoas, Descoberta de serviços, Padrões de mobilidade

Abstract

Energy saving plays a key role, not only for the growing need of cost reduction, but especially by the expected significant reduction of natural resources. The knowledge of the location and patterns of mobility of people inside buildings can help reduce consumption, making it possible, for example, to adjust the air conditioning and lighting to the real occupation of the building. We propose a system capable of counting people and learn their patterns of mobility inside buildings by giving the ability to integrate several technologies. This thesis also contributes to a multi-functional monitoring platform by integrating a service discovery module. In this document, we also analyze the state of the art technologies and protocols used for service discovery and locating people.

Keywords: People Counting, People Tracking, Service Discovery, Patterns of Movement

Contents

List of Tables

List of Figures

List of Acronyms

SDS Secure Service Discovery Service . . . 3

UPnP Universal Plug and Play . . . 3

PAN Personal Area Network . . . 15

P2P Peer-to-peer . . . 7

GSD Group-based Service Discovery . . . xiii

Allia Alliance-based Service Discovery . . . 8

LAN Local Area Network . . . 4

MEMS Micro-Electro-Mechanical Systems . . . 17

UMP User Mobility Pattern . . . 18

UMH User Mobility History . . . 18

MT Mobile Terminal . . . 18

UAP User Actual Path . . . 18

SPI Serial Peripheral Interface Bus . . . 8

I^2C Inter-Integrated Circuit . . . 9

WLAN Wireless Local Area Network . . . 4

OWL Web ontology Language . . . 36

RSS Radio Signal Strength . . . 27

UMP User Mobility Pattern . . . viii

API Application Programming Interface . . . 38

API Application Programming Interface . . . 33

YAML Yaml Ain't Markup Language . . . 36

XML Extensible Markup Language . . . 36

JSON Javascript Object Notation . . . 36

OS Operating System . . . 35

DIO Digital Input/Output 35

USB Universal Serial Bus xiii

YAJL Yet Another JSON Language 36

RFID Radio Frequency Identification 39

GSM Global System for Mobile Communications 39

Chapter 1

Introduction

1.1 Motivation and Objectives

Energy saving plays a key role, not only for the growing need of cost reduction, but especially by the expected significant reduction of natural resources. The knowledge of the location and patterns of mobility of people inside buildings can help reduce consumption, making it possible, for example, to adjust the air conditioning and lighting to the real occupation of the building. The cost to determine location should be inferior to the savings. Nowadays, people carry communication devices. Those could be used to infer their location and to assess their mobility patterns. Current systems tend to use only one technology to achieve location and user mobility patterns. Most systems are even intrusive by needing people to carry specific devices or installing applications on their mobile devices. Regarding the data acquisition, these systems and services often need to acquire information from heterogeneous sources, i.e. sensors, wireless interfaces etc. In addition, these sources can have different properties like periodicity, information push or polling, among others. This means that a generic sensor module is needed to support any kind of information source and be adaptable to different properties.

Considering this, we propose a system that makes use of several communication technologies and terminals to be capable of counting people and learning their patterns of movement inside buildings to estimate and reduce energy consumption. This system will then be installed at Instituto Superior Técnico – Taguspark's campus as part of a project concerning Green IT in an association with MIT – Portugal. This system will also function as a use case of a multi-functional monitoring platform. This platform handles all communication between services and abstracts the applications from the problems that arises from that. Furthermore, adds filters that aide the application to respect some requisites like synchronization, aggregation of packets and network monitoring. However, we felt that a platform of this kind, especially for pervasive environments, should also provide service

discovery. Services and clients need to communicate between themselves to obtain information even though they don't know each other, what they are and how they should communicate. Therefore a service discovery module will be integrated into the platform which takes into account the scalability of such systems and the numerous types of services existent.

1.2 Structure of the document

This document is organized as follows. Section 2 provides an introduction to current protocols and previous projects related to any of the areas of this work. Section 3 introduces the designed architecture of the system as long as all its components. In section 4 the corresponding implementation is discussed and the specific developments made for installation of a prototype in IST-Taguspark. Section 5 evaluates the implemented solution. Finally, section 6 concludes this document by summarizing its most relevant points.

Chapter 2

State of the art

This section is divided in five major areas that are addressed by this thesis. We will start by describing early, and most adopted, protocols in service discovery as some more recent research that better deal with new challenges. In section 2.2 we approach techniques and projects that deal with data acquisition from heterogeneous interfaces. We then enter the use case domain and describe the main technologies used for counting and tracking of people as some projects in which they are used in sections 2.3 and 2.4. We end this section by expressing current methods used for inferring data patterns, specifically patterns of movement, in section 2.5.

2.1 Service Discovery

2.1.1 Standard Protocols

Service Discovery is a widely investigated subject. It is defined as a method of networked servers to advertise their services, lookup for services provided by other nodes, select the most appropriately matched services and eventually invoke them or learn how to[44]. We can divide service discovery protocols into two types: directory-based and directory-less. In directory-based protocols the nodes can act as a server (providing one or more services to clients), a client (requesting services) or a service directory where servers register and clients lookup for services. In directory-less there are no intermediates between service providers and clients. Usually, the service providers broadcast service advertisements and clients broadcast service requests. Currently, the most adopted service discovery protocols[54] are Ninja Secure Service Discovery Service (SDS)[14], Sun Microsystems' Jini [46], Microsoft's Universal Plug and Play (UPnP)[1], Apple's Bonjour and Avahi [2], Salutation[3], IETF's Service Location Protocol (SLP)[43] and Bluetooth's SDP[4].

[1] `http://www.upnp.org/` accessed on November 18th, 2010.
[2] `http://avahi.org/` accessed on November 18th, 2010.
[3] Salutation was disbanded on June 30th, 2005.
[4] `http://www.bluetooth.com/` accessed on November 18th, 2010.

Ninja SDS is a hierarchical directory-based protocol that focuses on the security and privacy in service discovering providing methods for authentication, authorization and confidentiality. This protocol does not provide service invocation methods and service naming. Ninja SDS supports discovery scope to be user role based, context based or network topology based. User role discovery scope is based on administrative domains. Users have access to certain services based on their role in the environment. Context based is high-level information such as temporal, spatial, and user activity. In this protocol, context based discovery scope is supported by spatial information. Network topology based discovery scope is limited by the underlying network reachability, in this case, the Local Area Network (LAN).

Jini is a service discovery architecture to be performed among java-enabled devices. Jini is directory-based and can have a flat or hierarchical infrastructure. It provides mechanisms for service communication through RMI and download-able java code. The discovery scope is similar to SDS. Jini also offers service and attribute naming for appropriate matching.

UPnP was designed for service discovery in home and enterprise environments. It is mostly used as a directory-less discovery service but it may use control points that act as service directories. The devices advertise their presence through multicast and, when queried, present capabilities through XML for service description. UPnP provides service invocation mechanisms through XML, SOAP and HTTP. It also provides secure service discovery and access through UPnP security.

Bonjour and Avahi are both Domain Name System - Service Discovery[12] based. These are directory-less protocols, this goes against the DNS architecture. Thus, a variant of the DNS is used: Multicast Domain Name System[13]. The devices multicast their DNS queries specifying the service type wanted, the domain to search for and the preferred communication protocol. Servers respond through DNS service records. Servers can also announce their services when arriving at the network making other devices aware of their presence. The service records are kept in client's cache for a limited time by which new queries are needed.

Salutation is a directory-based service discovery protocol designed for home and enterprise environments. The main difference to other protocols is the fact that it may operate over different network technologies like infrared, Bluetooth, Wireless Local Area Network (WLAN), etc. Devices communicate with each other through a component called Salutation Manager that abstracts the technology communication through different transport managers. This component integrates the service discovery functions of a client, service, and directory into one component. This enables one device to act according to different roles in different situations. Devices have their capabilities expressed as

attribute sets. In terms of security, salutation supports only password-based authentication.

The Service Location Protocol (SLP) is one of the most adopted protocols in commercial products [44]. SLP only specifies the service discovery and not the service invocation. SLP describes services through unique URLs and a set of attribute-value pairs. This protocol can operate both in directory-less and directory-based. In the first case, there are only User Agents (UAs) on the clients and service agents on servers. The communication between them is performed through multicast. In the later case, directories are represented by directory agents (DAs). These agents, when entering the network, multicast a beacon in order that SAs register their services in the directory. If UAs do not know any DA when preparing a query, they multicast queries and all receiving SAs with matching descriptions respond using unicasts. Otherwise, they simply send the query unicast to the known DAs.

The Bluetooth SDP is a service discovery protocol specifically for Bluetooth-enabled devices. Contrary to all the other protocols, this protocol only supports query-based service discovery. It does not address known service caching or service access. This protocol is only able to discover services at a single hop range, thus, is limited in discovery capabilities to the neighboring nodes. It provides two methods of discovery: service searching and service browsing. In the first method, the desired service attributes are contained in the request, these are matched with available services in the service provider at range and the result is then returned. The second method supplies a list of all the services provided by the device being queried.

In Table 2.1 we compare these service discovery protocols by the following properties: service and attribute naming, method of service discovery and registration, type of infrastructure, the discovery scope and the security properties support. This table was based on the F. Zhu work [54].

2.1.2 Service Discovery in Pervasive Environment

These prior protocols are not suitable for pervasive environments [54]. In pervasive environments, networks are often unadministered and connections are mainly made through wireless ad-hoc networks and wireless infrastructure-based networks. Considering this, new challenges arise: pervasive computing nodes may move and thus affect service availability; there are frequent disconnections between nodes and the path to each node may change; the communication channel can also suffer from propagation issues, thus decreasing network performance like data rate, delay, jitter, etc. As a result, centralized directory-based service discovery, like SDS, Jini, Salutation and SLP, is not ideal since in pervasive computing we cannot consider that a node is always reachable by all the other nodes. On the other hand, directory-less service discovery, like UPnP, Bonjour, Avahi and SLP (directory-less version) make too much use of broadcast and multi-cast which can limit the scalabil-

Table 2.1: Comparison of early service discovery protocols

	Service and attribute naming	**Discovery and registration**	**Service Discovery infrastructure**	**Discovery scope**	**Security**
Ninja SDS	Not Available	Query and announcement based	Hierarchical Directory-based	User role (administrative domain), context (location), and network topology (LAN)	Authentication, Authorization, Confidentiality, Integrity, Non-repudiation and Service privacy
Jini	Template-based and predefined	Query and announcement based	Flat or Hierarchical Directory-based	User role (administrative domain), context (location), and network topology (LAN)	Authentication, Authorization and Integrity
UPnP	Template-based and predefined	Query and announcement based	Directory-less	Network topology (LAN)	Authentication, Authorization, Confidentiality and Integrity
Bonjour	Template-based	Query and announcement based	Directory-less	Network topology (LAN)	Authentication and Integrity (DNS Security Extension)
Salutation	Template-based and predefined	Query and announcement based	Flat Directory-based	Network topology (LAN)	Authentication and Authorization
SLP	Template-based	Query and announcement based	Directory-based or Directory-less	User role and network topology (LAN)	Authentication and Integrity
Bluetooth SDP	Template-based and predefined	Query based	Directory-less	Network Topology (single-hop ad-hoc network)	Authentication, Authorization and Confidentiality

ity, overload the network and limit the reachability of these systems. Nevertheless, these protocols were a good solid base for new protocols, better fit to pervasive environments. We will now analyze more recent work that tries to deal with the new challenges.

Based on the previous statements, we are looking for discovery service protocols that are either directory-based with a distributed architecture or directory-less but with flooding control.

Directory-based

In directory-based service discovery there are three types of distributed directories: Backbone-based, Cluster-based and Distributed Hash Table-based.

Backbone-based : In backbone-based architectures there is at first the need to form the backbone. Current research, use Minimum Dominating Set algorithms to form the backbone [29, 38]. Servers advertise their services to one or more directories. When clients execute a request to a certain backbone member, if the request isn't satisfied locally, it propagates the request to other backbone members. Therefore, services are globally discoverable. In [38] they improve the propagation of requests by forwarding them to nodes more likely to know a service provider that matches the request (backbone nodes exchange directory profiles).

Cluster-based : In cluster-based approaches service providers are grouped in clusters based on semantic proximity [28], physical proximity [27] or by mobility patterns [39]. Clusters exchange summaries about all the services they know. These protocols also provide global discoverability since they forward requests that cannot be satisfied locally to neighboring clusters. These types perform better in terms of messaging overload in different scenarios. Semantic proximity is best for accessing services of a particular type with specific attributes since similar services are grouped together; physical proximity is used for instance for users to discover services that are available in their vicinity; grouping servers by their mobility patterns has the benefit of always existing a server capable of responding to service requests and thus, applying energy-efficient algorithms like only one member of the cluster being awake each time [39]. However, these approaches are not ideal when aiming at a generic platform that can support different scenarios.

DHT-based : There is also some research on distributed hash table-based service discovery [40, 41]. In these solutions, the service discovery network is divided into geographical regions. Each region has a set of keys associated that represent services. For each existent service, the resulting key is stored in the corresponding region (by representing nodes). When nodes request a service they use the same hashing function and thus find the location where its description is stored. Therefore, global discoverability is provided.

These directory-based architectures simplify service advertising and service requesting. However, they imply additional communication costs in the network for maintaining the directory structure [44]. From electing the directory nodes to exchanging data among nodes and preserving consistency, this adds a great deal of complexity to the system when compared to directory-less architectures. Moreover, when these issues are not well addressed, it may be generated too much traffic to maintain consistency or it may decrease service availability by inconsistencies in replicated directories.

Directory-less

In directory-less architectures the major challenge faced is the flooding control of broadcast or multicast advertisements and requests. The main techniques used for dealing with this issue are[44]: Scheduling and prioritization of service advertisements; Advertisement range bounding/scoping; Selective, probabilistic, and intelligent forwarding of advertisements and requests; Peer-to-peer (P2P) information caching; Intermediate node responding to service requests.

In [7], service providers periodically advertise their services along with services that they became aware of to the neighbor nodes. When a provider receives an advertisement that contains its own services, it postpones advertising its services and back-offs for a fixed amount of time since the

nodes in his vicinity already know his service.

In the GSD[10] and Alliance-based Service Discovery (Allia)[37] protocols, advertisement range bounding is used by specifying the number of hops by which the advertisement message will be dropped. For nodes to become aware of all the services provided in the network, these protocols also implement P2P information caching: Nodes merge services learnt from previous advertisements and re-advertise them along with their service using range bounding. Regarding service discovery, these approaches employ selective forwarding to avoid network flooding. When a node does not match the service request it forwards the request only to neighbors that are known to match the requested service or host similar services.

In [33], intermediate nodes are allowed to answer service requests of services they do not provide but are aware of. However, this may decrease the number of discovered services since intermediate nodes may not be aware of all the corresponding services that match the request and therefore respond with a less suitable service provider. To deal with this issue, the authors propose that when answers come to a service requestor from different servers and different paths, intermediate nodes and servers along those paths be updated to become aware of all the services that were returned to the requestor.

Considering the goals of the proposed platform, we consider that directory-less protocols are more suitable to a platform that can be used in very different scenarios, i.e., nodes may have very low processing capabilities, and therefore the complexity imposed by directory-based protocols is not indicated. From these directory-less protocols we highlight the GSD protocol for the global service awareness based on P2P information caching, since this method is more efficient than the ones seen in [7] and [33]. For service discovery they also implement selective forwarding which may not be as effective as in [33] but this protocol may not know the best corresponding service to the request. The choice of GSD facing Allia is due to the fact that the former is much more evaluated than the last.

2.2 Data Acquisition from Heterogeneous Interfaces

In order to analyze how to implement a generic sensor interface that supports heterogeneous data sources we will first look at low-level protocols commonly used for communication with embedded sensors.

Serial Peripheral Interface Bus (SPI) is a synchronous serial bus data link that operates in full duplex [26]. It has Master/Slave communication relationship, in which the master initiates the data

frame. Slaves can be thought of as input/output devices of the master. Data may be transfered in both directions simultaneously. It is also possible to have multiple slaves as the master generates slave select signals. Master/slaves must use the same parameter pair values to communicate. SPI does not have an acknowledgment mechanism to confirm receipt of data neither offers flow control.

Another low-cost, low-speed communication protocol is the Inter-Integrated Circuit (I^2C). As the SPI, it also provides good support for communication with slow peripheral devices that are accessed intermittently. I^2C is easier to implement and require less hardware resources than SPI when more than one slave is involved due to its built-in device addressing. However, when point-to-point communication is used it creates more overhead for the same addressing feature, consequently, achieves lower data rates than SPI.

After analyzing low-level protocols for communication with sensors we will now refer projects that implemented an abstraction layer for generic data acquisition from different types of sensors.

In [9] they define a generic sensor interface for robot simulation and control. They refer generic as the ability to handle different sensors. The architecture used in this work can be seen in Figure 2.1. The interface is divided into three main components: Message Encoders/Decoders, external and internal message handlers and communication components. The first are responsible for transforming messages between users/sensors and message handlers. The message handlers are responsible to handle requests by users and route requests to sensors. The communication components are responsible for the communication protocol between the interface and the sensors or the interface and the user applications.

The [6] work is a platform that supports the integration of different types of location sensors to achieve people localization. This platform may receive data from one or countless sensors and merges the location information from the different sensors for better accuracy. This feature will be further analyzed in Section 2.4.9, we will now look into the support of the multiple sensor integration. The platform architecture may be consulted in Figure 2.2. We are interested in the Sensor Plugins and Event Abstraction Layer modules since these are the modules that function as an interface between the sensors and the core of the platform. In their context, a sensor may be a physical device or an external application. A sensor plugin exists for each type of sensor that is to be integrated in the platform. The function of the sensor plugin is to transform sensor location information into an abstract representation of a sensor event (Event Abstraction Layer). Sensor plugin can be of two types: synchronous or asynchronous. Synchronous sensors provide information only when they are prompted to (pull semantics), for instance, querying bluetooth for in range nodes. Asynchronous sensors provide information at irregular intervals, usually when a certain environment event occurs

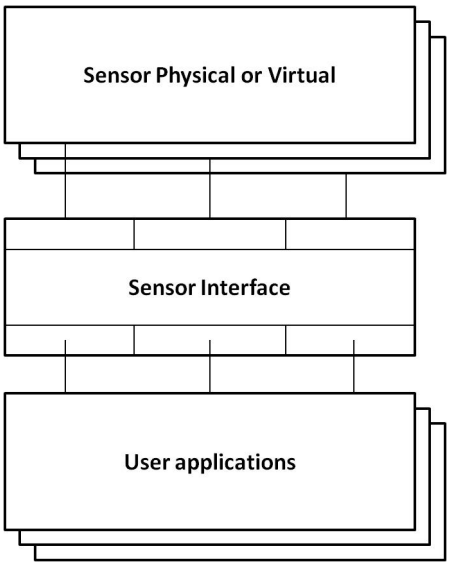


Figure 2.1: Generic sensor interface architecture [9].

(push semantics), for instance, when an infrared beam is obstructed it triggers an event. These asynchronous sensor events are stored temporarily in an internal event buffer until a reading is made.

2.3 People Counting

In this section we will describe the main technologies and methods used for people counting as some real implementations.

2.3.1 Infra-red

When using this technology to count people there are two main methods used: through infra-red beams or infra-red thermopiles [24]. In the first case, the beams are positioned horizontally at door entries and when the beam gets broken it means a person is passing and therefore is counted. This simple approach doesn't take in account directional flow and so a dual beam approach is often used. Consequently, information about entrances and exits is available. The accuracy of this approach is limited, especially when high volume flow or uninterrupted traffic occurs. It is also possible to position the beams vertically which can increase somewhat the accuracy. The second method is based on people detection through body temperature. This method has the advantage that these sensors are passive and so do not require to radiate energy to the environment, additionally, thermopiles measure heat and therefore are not affected by direct lighting as infra-red beams [24]. On the other hand, this is a more complex detection algorithm as the human body does not emit temperature uniformly and can vary with different clothes and time of day [32].

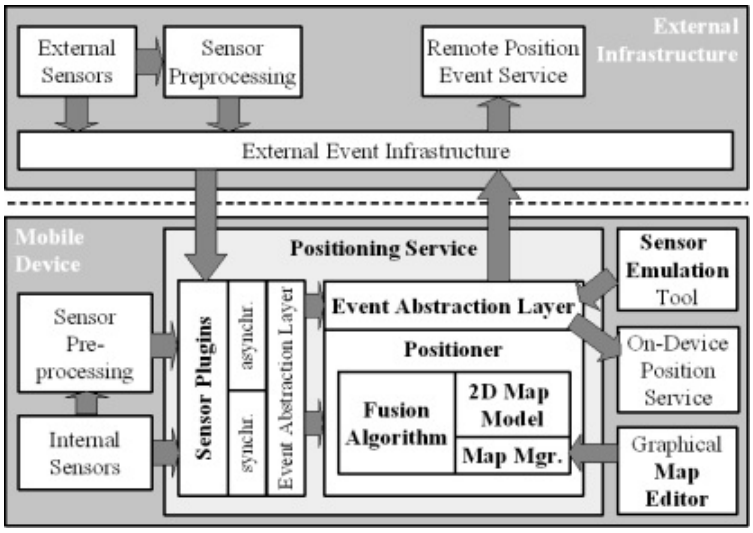


Figure 2.2: Architecture of the probabilistic positioning service [6].

2.3.2 RFID

Radio frequency identification is a wireless communication technology for data exchange between a reader and a tag. Each tag has their own ID that is emitted to the reader. RFID tags can either be active or passive [35]. Passive tags operate without the use of battery. They reflect the RF signal received from the reader and add information by modulating the reflected signal. These tags are much cheaper and have a virtually unlimited lifetime. However, they have a limited range. Active tags contain a battery to power the transceiver and so can achieve much higher ranges. Besides being more expensive, its battery life is of course limited although it can go up to several years. The main purpose of this technology is for the identification of objects, for instance, commercial products, cars and others. However, it may also be used for people counting, location and tracking. Specifically for people counting purposes, when considering entrances and exits of divisions, it makes sense to use low-range passive RFID tags not only for their lower cost but also to get information more accurately from readers situated at door entries. The main setbacks of using this technology for people counting are the requirement of people to carry or wear RFID tags, and also the privacy issues that must be considered since an RFID tag can hypothetically identify the person.

2.3.3 Vision-based

People counting can also be achieved through vision-based technology. Recurring to video cameras it is possible to detect people passing through doors and gates and therefore count them [11, 52]. The detection is usually based on background subtraction, discovery of people's shapes, skin tones, etc. This type of detection faces multiple difficulties. People wear different types of clothing, gloves, use accessories like purses and umbrellas and may enter facilities "merged" with other people.

Furthermore, lighting changes and weather conditions, when outdoors, worsens detection. When these challenges are correctly faced, the accuracy of these systems can be very high. However, this creates complexity in the system and video processing by itself is already computationally heavy.

2.4 Location and Tracking of People

In this section we will describe the main technologies used to achieve location and tracking of people as some localization systems.

2.4.1 Location systems properties

Before describing the technologies that make location possible, we will first express the different properties of these systems [22]:

- **Localization Techniques**: In terms of localization techniques there are three major methods used. **Triangulation**, **Proximity** or **Scene Analysis**. The triangulation can be of two types: **Lateration** - which measures the distance between multiple known points - or **Angulation** - which measures the angle or bearing relative to points with known separation. Proximity measures closeness to a known point. Scene Analysis examines a view from a known point.

- **System topologies**: The location systems can either be for a device to locate itself or for a remote system to locate devices or objects. From these two hypotheses we can infer four types of system topologies [45]:

  Self-positioning - In a pure self-positioning system the computation of position is made localy. In this topology the user's privacy is assured since only the device knows its own position.

  Remote positioning - In a remote positioning system the computation of objects location is made on the infra-structure. Therefore privacy issues may arise; however, the major advantage of these systems is the fact that the objects being located when mobile devices are required, the computational and power demands on these devices are decreased.

  Indirect remote positioning - This is a self-positioning system that transfers measured data to a remote site.

Indirect self-positioning - This is a remote positioning system that transfers the measured data from the infra-structure to the mobile unit.

- **Physical position and symbolic location**: Location systems can provide physical and symbolic information. For instance, physical location can be GPS coordinates. On the other hand, symbolic location supplies abstract information usually about proximity to certain known points, i.e. in the bedroom or driving near Leiria.

- **Absolute versus relative location**: Absolute location is used when every device has the same reference for all located objects. For example, a GPS coordinate refers to the same place regardless of where we're looking from. In relative location each object can have its own reference and obtain a position relative to itself. For instance, in a treasure hunt where the spot has a transceiver, the hunters may locate the treasure's position relative to them selves.

- **Accuracy and precision**: These metrics measure the performance of location systems. Accuracy is the margin of error in measurements, usually in meters, precision is the percentage of measures within that accuracy. For example, GPS has an accuracy of 5m with a 99% precision.

- **Scale**: The scalability of a location system is measured by the coverage area per unit of infra-structure and by the number of objects the system can locate per unit per time interval.

- **Recognition**: The ability to recognize devices by an identification mechanism. This property is used by systems that need to take action based on the device located. For example, the use of RFID tags in cars to automatically pay tolls in highways, it is necessary to identify the device to charge the payment to the car owner.

- **Cost**: The cost of a location system may be divided into three types. The Time costs associated with installation and administration needs, the Space costs involve the amount of infra-structure units and their size and shape and finally the Capital costs related to price per device and infra-structure unit and also the salaries of support personnel.

- **Limitations**: Location systems often have limitations in certain environments. For example, GPS has limited signal indoors and the devices are not able to get a fix on satellites.

2.4.2 GPS

The GPS technology is currently the most used localization system worldwide [15]. It provides physical absolute self-location through GPS devices that receive Satellite signals. Therefore, the computation of the location is made on the device which makes recognition not possible. GPS is highly scalable given that twenty-four satellites cover location all over the world, however, the infra-structure is highly expensive and each person's GPS device cost is fairly high. This technology also has a high value of accuracy (1 to 5 meters) and precision (95% to 99%).The main setback of this technology besides the cost is the need to have line of sight to at least three satellites to compute a position due to the triangulation method. This is generally not a problem for outside environment but indoors the signal from satellites is usually too weak.

2.4.3 Infra-red

The use of Infra-red technology for localization, currently, is mostly used in vehicles to achieve pedestrian detection [5, 16, 32]. This can be used either to improve pedestrian's safety by warning the driver (passive safety) or activating the ABS (active safety) or in automated robotic vehicles to avoid pedestrians. The main method used to accomplish pedestrian detection is through scene analysis by processing infra-red images to identify human shapes, size, aspect ratio, temperature and others[5]. The challenges it faces are the same as with people counting and it may only detect people and not objects. On the other hand, this is a low-cost technology, there is no recognition of the detected people, thus, privacy is not an issue and there is no need for the use of specific devices for people to be detected.

2.4.4 RFID

Localization and tracking is also possible with the RFID technology. However, in order to obtain high accuracy it is required that RFID tags always be at range in the pretended localization area. It is not viable to have RFID readers spread with high density in the localization area because of the cost of these devices and the infra-structure needed for this purpose. Therefore, RFID passive tags are not usually considered to achieve positioning. The typically methods used are triangulation and location sensing through reference tags. The first case is based on the computation of received radio signal strength from different readers [23]. This method experiences low accuracy, when indoors, due to the propagation problems of radio frequency signals such as reflection and multi-path. The second method is also based on received signal strength but requires less RFID readers since it obtains position based on reference RFID tags located on know points [34]. The nearest RFID reference tags are determined based on their received signal strength compared to the signal strength of the moving RFID tag,the location is then computed by triangulation but based on the location of the closest reference tags. This method dynamically adapts to propagation conditions thanks to the

reference RFID tags and the accuracy is proportional to the density of these devices in the location space. The main disadvantages of using RFID technology as described in 2.3.2 are the requirement of people to carry RFID tags to be located and the privacy issues regarding the identification of the person being located.

2.4.5 Wi-Fi

The Wi-Fi (IEEE 802.11 family) technology is widely embraced. It is currently on our notebooks, netbooks, PDAs and Smartphones and is available in buildings, hotels, cafes and others. This makes the Wi-Fi technology very attractive to use in localization systems because it enables localization with lower costs as the infra-structure is already present in most places. The major techniques used for localization through Wi-Fi are lateration and proximity, either by RSS (Received signal strength) or cell based positioning. Since the Wi-Fi is generally used indoors, when the RSS method is used, reflection and multi-path issues arise, therefore the accuracy is affected and can go from 3 to 30 meters [45]. When the cell based positioning is used, the location is made by either querying access points for connected clients through SNMP or by getting this information from RADIUS authentication system, the accuracy in these cases is fairly low tough because it depends on the size of the WLAN cell which can go from 50 to 300 meters [25]. This technology can provide either physic or symbolic position and the location computation can either be on the device or in the infra-structure. The scalability of these location systems is limited due to the infra-structure needs. Thus, this technology is more commonly used indoors.

2.4.6 Bluetooth

Bluetooth is a wireless communications technology standard. It is best suited for data transfer through Personal Area Networks (PANs), however, it can be used for localization systems. Bluetooth is a low-cost technology and is currently built-in in most of our mobile phones which makes this technology interesting to use for people localization. This technology allows self and remote positioning. In self positioning the method most commonly used is through received signal strength of beacons from one or multiple known points like in [53]; The main disadvantage of this approach is that specific software on the device is needed to perform the positioning. In remote positioning there are two methods used, connection-based and inquiry-based[36]. Connection-based systems measure device signal strength while connected to Bluetooth access points [2]. This can be intrusive to the owner since it establishes a connection at least once every time it is on range. Furthermore, it has limited scalability due to the Bluetooth limitation of seven simultaneous connections [21]. Inquiry-based positioning refers to periodically inquiring for in range devices and then store signal strength and compute location [36] or assume cell-based positioning [19]. In either case the main disadvantage is the fact that the devices must be in “visible” mode to be detected. Also, when

in indoor environments, signal strength can be affected by many factors like multi-path and reflection which can difficult triangulation or distort cell shapes and consequently lower the accuracy.

2.4.7 GSM

It is also possible to achieve localization through GSM. Some of the applications that were made with this type of localization are: the E911[5] , a service that automatically locates the person that is calling the emergency number and forwards the call to the closest station; services made available by mobile communications operators to enable parents to keep track of their children's location or companies to track their fleet's location; or even to obtain course grained information about user mobility like stationary, walking or driving [42]. In the first two examples, the localization is made by lateration. However, the accuracy of these systems are very low (150m to a few kilometers) due to the coverage area of a single antenna and the various distortions the signal can suffer, moreover this information is only available to the GSM service providers. The computation is made in the infra-structure and it is able to recognize the device. The last example analyzes GSM traces to achieve course grained information about self-mobility. It is also possible to obtain self-positioning using GSM traces. In [30] the GSM cell IDs are retrieved from the associated connections between the cell towers and the mobile GSM device. These IDs are then matched in a database and the corresponding physical positions of the cell towers are obtained. The self-position is then computed by lateration based on the location of the towers and the received signal strength. This has the disadvantage of needing an application in the cell phone to compute the positioning and the need to have a pre-filled database with the location of GSM cell towers. The main advantages of using this technology for localization are the current coverage and penetration of mobile phones which avoids the need to acquire any specific devices.

2.4.8 Vision-based

Localization can also be feasible through video. By processing video images it is possible to infer people's shapes and movement and pinpoint their position resorting to scene analysis. Regarding the infra-structure, most buildings already have surveillance cameras which can reduce the cost of deployment. Vision-based localization can provide physical or symbolic position. This type of localization is very accurate, in order of centimeters to a few meters[3]. On the other hand, video processing is very demanding in terms of computational capabilities which can increase the cost of localization processing units [32], furthermore it has a limited scalability as a result of this intensive computation and also the infra-structure needed to connect to video processing units. Another challenge it faces is the need to merge cameras vision to a single plane due to the overlay of camera vision or the lack of sight of certain areas [31].

[5] `http://www.fcc.gov/e911` visited on December 20th, 2010.

2.4.9 Accuracy and precision improvement

In order to improve positioning and tracking accuracy, many localization systems integrate multiple location sensor sources. To achieve this integration, systems need to correlate location data from different sensors and merge this data to obtain a more accurate single location result, this is usually called Sensor-Fusion. In [6] they provide a platform capable of integrating unlimited location sensor sources of any type (System architecture can be seen in Figure 2.2). They resort to probabilistic methods to compute the device location. Each sensor type has a corresponding module that provides the system with a set of possible locations represented as a probability distribution. These distributions are merged to compute a new probability distribution that is then matched with an existing map of the site in order to eliminate points that correspond to walls, obstructions, etc. Finally they implement an algorithm based on the previous known location of the device and the probability distribution to estimate the possible movement and therefore most likely location to where the person has actually moved. A more specific approach to sensor fusion is made in [48], where they aim at 3-D indoor localization with barometric sensor data, WLAN triangulation and building information. To compute a location they start by measuring the position floor based on altitude information obtained from a barometric sensor with the building information (sea-level altitude and height between floors). The floor's respective 2D map is then used to match the computation of the 2D location based on the RF signals (This is made through RSS triangulation based on the K-nearest neighbors algorithm [1]) .

There is also some work in localization systems that use filtering estimators to improve accuracy. In [4] they present a system for localizing people through vision-based technology only. Their main enhancement to usual vision-based localization is the implementation of an algorithm that tracks people based on the evolution of each pedestrian's movement through Kalman filtering[50]. Kalman filtering enables trajectory smoothing and thus, reduce estimation errors. Another approach combines a multi-sensor approach with a Kalman filter based fusion and tracking [17]. In this system both infrared camera and laser scanner technologies are used to locate people. Laser data and infrared-image processing are performed separately; each technology data processing uses it's own kalman filter, which means each sensor has a multi target tracking. Data fusion is then performed on a central fusion computer also based on a kalman filter. However, kalman filtering has been proven not to be the best solution because its main assumption, i.e. the linear model, is hardly fulfilled in real life [47]. In this same work they describe a system that achieves pedestrian tracking through the integration of WLAN-based location (K-nearest neighbors algorithm) with low-cost Micro-Electro-Mechanical Systems (MEMS) accelerometer and map information with particle filters [20]. Particle filters can deal with non-linear and non-Gaussian estimation problems. They compare the use of kalman filters facing particle filters and show that the accuracy can be substantially increased by using particle filters and further integrating additional information, in this case, walking distance and map information.

2.5 Identification of Patterns of Movement

Regarding patterns of movement the goal is to provide statistical description of typical pedestrian paths for a given time of day. Most research in mobility pattern matching use cell based (hexagons) division of the localization environment in order to register movement between different cells [8, 49]. In [8] they infer User Mobility Patterns (UMPs) from User Mobility Historys (UMHs). These UMHs are data structures that store the mobility history of a Mobile Terminal (MT). Each MT manages its own UMH and registers its own movement. A UMP is a sequence of cell ids, that represent its path, and a corresponding expected cell entry time. A UMP is created everytime a new trajectory is made or at different time intervals. Everytime the MT registers a movement, it verifies which UMP matches its own path. These UMPs eventually become more used and it becomes possible to deduce the most made trajectories at different times of day.

Another approach is to use sequential pattern mining to obtain UMPs from User Actual Paths (UAPs)[51]. They use a directed graph that interconnect neighboring cells in the coverage region which represents the fact that a user can move between these cells. The mining is achieved by processing all UAPs and incrementing candidate paths. When users following a UMP perform random paths between the consecutive cells of the same UMP, these are considered as noise and the UAP is called corrupted. If the number of corrupted UAPs is high, then a pattern may be missed.

Chapter 3

Architecture

3.1 Introduction

This chapter will analyze the designed architectures for the proposed systems. We start by describing the requisites of the location system and the physical and logical topologies. After that, we described the platform used to support our system - the Multi-Functional Platform for Indoor and Outdoor Monitoring - and the service discovery component that we had to integrate on it. To finalized, we described the architecture of each one of the component of the location system. We finalize with a short summary of the chapter.

3.2 Location and Tracking System

3.2.1 System requirements

The main goal of this thesis was to create a flexible location and tracking system that could operate on different environments. This system must be able to infer user mobility patterns and, ultimately, supply the tools so that other applications reduce energy consumption.

The main requirements of the systems are:

- Be non-intrusive so that users can be located without their intervention in order to guarantee that their behavior is not affected by the system.
- Be scalable to different building types and sizes or open spaces, in order to allow location in a wide range of scenarios.
- Be flexible so that different types of technologies can be used to perform different types of functions, such as location, tracking and pattern movements detection..

- Be able to run on low cost nodes with low computational power. This way, a large number of nodes might be used, providing more accurate results.
- Provide the capacity to integrate different sensing technologies.
- Maintain the people's privacy in order to guarantee that no one can know the exact location of each person.

3.2.2 System overview

Considering the above mentioned requirements we created a distributed and modular architecture. The location system is composed by nodes that are spread in monitored buildings. Each node have a set of sensing devices that enable it to retrieve important information about people location in its neighborhood. There may be different sensing devices in different nodes, that may complement each other. Thus, Wifi technology may be used to perform long-range location, bluetooth to perform short-range location, infrared to count people and RF-IDs to track persons. At the end, the information gathered by each one of the nodes should be collected by a sink node and sent to a server to be stored.

An example physical topology of this system is illustrated in figure 3.1.

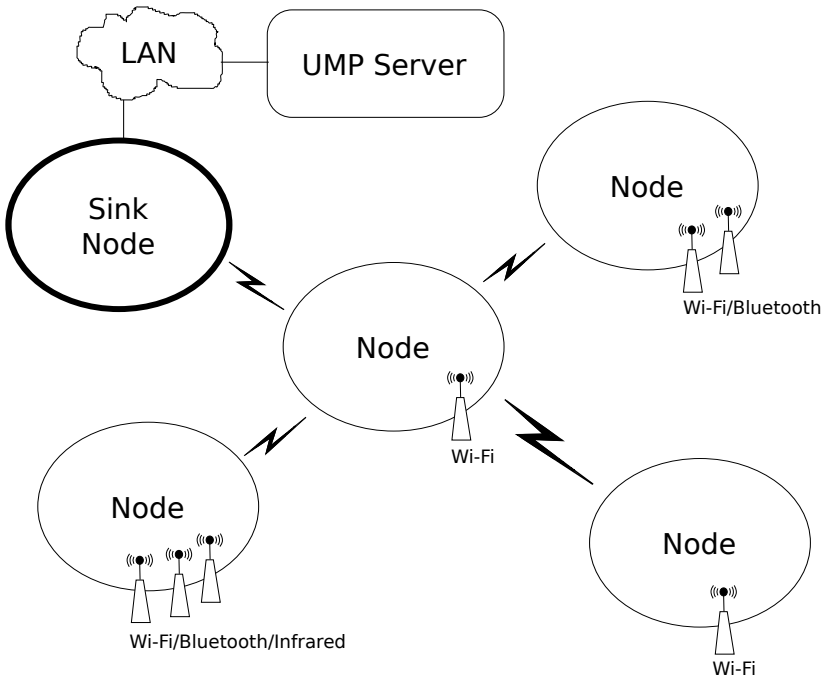


Figure 3.1: Example physical topology of the system.

In order to have a scalable network architecture, we used an hierarchical network structure, as illustrated in figure 3.2. The nodes in the bottom of the hierarchy perform the most low-level operations, such as handling hardware communication for sensing people, and the nodes in the higher levels perform high-level operations, such as grouping information.

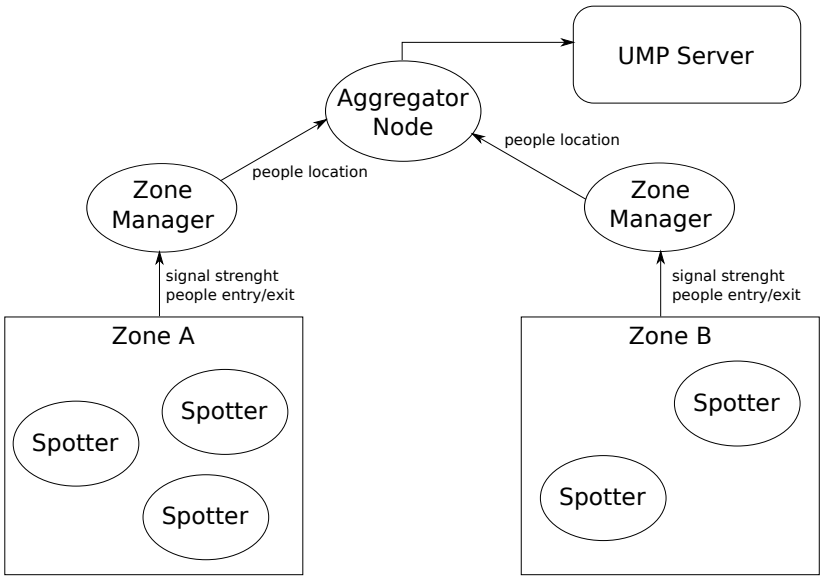


Figure 3.2: Logical topology of the system.

A building may divided in different zones, each one of them having a set os spotters that are exclusively associated to that particular zone. Each spotter retrieves signal strength and people entry/exit information from sensors and sends it to its own Zone Manager. The Zone manager collects information from spotters and use it to compute the physical or symbolic location of users within the zone. Each Zone Manager communicates with the Node Aggregator that is responsible for aggregating location information of all zones and transmit it to the User Mobility Patterns (UMP) server. This node also acts as sync node, as it is the node that is connected to the local area network (LAN). The UMP server is located on the LAN. The UMP server has virtually unlimited power and processing capabilities. It periodically receives the global state of localization and counting and stores the user mobility history. Based on this history, it continuously computes the user mobility patterns

Nodes announce their capabilities using the service discovery protocol that we developed in the Multi-Function platform. Furthermore, each node uses this platform to handle the data transport and synchronization. Each node can operate more than one component.

The next sections will describe the specific architecture of the the Multi-Function Platform and of each one of the components that have been described.

3.3 Multi-Functional Platform for Indoor and Outdoor Monitoring

3.3.1 Platform overview

The multi-functional platform for indoor and outdoor monitoring is a modular platform that provides a set of features to applications through a dynamic linked library[18]. It integrates a group of traffic interceptors that ease the development of traffic management techniques. Furthermore, it allows the configuration of the platform in order to adapt to each scenario.

For the purpose of this thesis we consider a high-level view of the platform and enhance the already implemented services for it. Therefore the architecture of a node using the platform is represented in figure 3.3. The applications that use the platform, contain a communication handler layer that performs the communication between the application and the platform or directly to another local application. Inside the platform layer are the different services provided. The modules represented with a continuous stroke are the ones used by this thesis. The service discovery module represented in the figure with a filled background is part of the work of this thesis and will be discussed in the next section. The use of this platform is justified by the abstraction of the applications to the underlying communication complexity and issues, the use of synchronization in order to easily resolve issues concerning delays or ordering of packets and to discover services based on descriptions even in complex dynamic networks.

3.3.2 Service Discovery

Service discovery is starting to assume a more important role in today's networks since networks are becoming widely available in our houses, streets, pockets and even on our clothes. These networks are more and more dynamic and automated. For this reason we felt the need to integrate such a service into the platform described in section 3.3. To integrate such a service in the referred platform without tampering with it's own architecture, this module is no more than an application in the perspective of the platform. However, it functions as an add-on to the platform, since other applications make use of it through an API. The resulting conceptual conjoined architecture can be seen in figure 3.4.

The service discovery API consists of a library and a set of simple interaction functions that will activate the service discovery module according to the application requests:

- RegisterService - To let the module know the start of a service providing. An application can

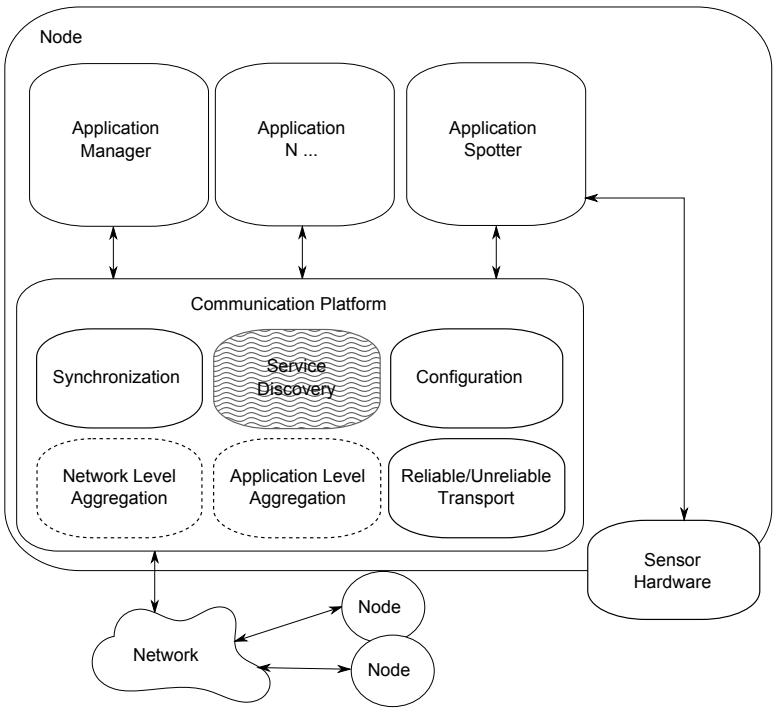


Figure 3.3: High-level architecture of the multi-functional platform.

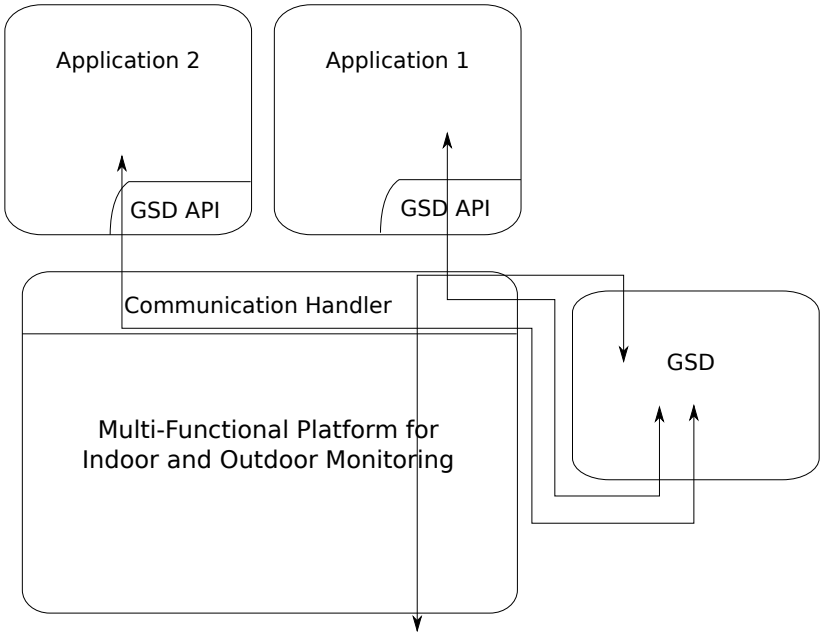


Figure 3.4: Integration of the GSD module in the platform.

provide more than one service and should register each one.

- StopService - To let the module know the conclusion of a service providing. As the register, it must declare the conclusion of each stopping service.
- RequestService - To an application request a desired service to the module.

The service discovery module architecture is based on a well-defined protocol, namely, the GSD [10]. For the reasons explained in section 2.1, the GSD is a protocol well suited to pervasive environments and mobile ad-hoc networks. It has a decentralized design approach and implements several techniques to better adjust to this type of environment, for instance, peer-to-peer caching of service advertisements and group-based intelligent forwarding of service requests. We will now do a short description of this protocol and our specific use and for more detailed information please consider reading the *Dipanjan Chakraborty* work [10] .

The architecture of the service discovery add-on to the platform can be seen in figure 3.5. The main components that implement the GSD protocol are:

- Advertiser
- Service Cache
- Service Matching
- Request Routing

First of all, services are described using a complex structure and are categorized into several groups based on class/sub-class hierarchy and properties. For instance, a video-camera could be classified under *Service / Hardware / Surveillance / Video-Camera*. This makes it easier to match requests to services provided and better forward the requests to the service group. The advertiser is a component that periodically advertises the list of known services to it's vicinity nodes. An advertisement consists of the following:
$< Packet-type, Source-Address, Service-Description, Service-Groups, Other-Groups, Hop-Count, Lifetime, ADV\_DIAMETER >$
The packet-type contains not only the type advertisement but also the broadcast-id that along with the source-address, identifies an advertisement and as such can detect duplicate advertisements. The Service-Description and Service-Groups contain information about the local services and their corresponding service groups. The Other-groups contains information of non-local service groups. This list is built based on data from the transmitting node cache memory. The Other-Groups field makes it possible for group information to eventually propagate to all the network. The $ADV\_DIAMETER$ and Hop-Count tells the receiver of the advertisement whether it should retransmit the advertisement or not. If positive, it increments the hop-count and transmits. Lifetime is

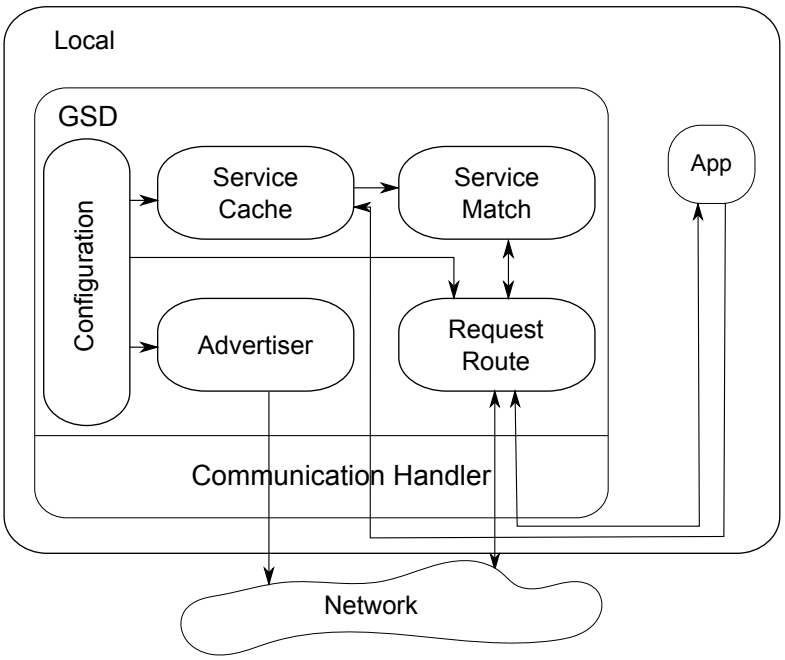


Figure 3.5: GSD module architecture.

the amount of time for which the info should remain in the node cache. Therefore, the service cache component stores the locally provided services as long with all services still active that it received an advertise from. Each entry of the service cache consists of the following:

$< Source - Address, Local, Service - Description, Service - Groups, Other - Groups, Lifetime >$

The flag Local is to identify whether this entry is referent to a local or remote service. Lifetime contains the remaining lifetime of this entry in the cache. All the other fields are the same as the received advertisement. When an application requests a service, the request routing component first searches its own cache to find a match and, in case of it does not find a match, it routes a request throughout the network. Based on the Other-Groups information in each node, the request is selectively forwarded to the node most likely to know the service. The request ends when the service is found and is then reversely routed to the request origin. This is done by maintaining a reverse-route table in every node that contains the following:

$< Source - Address, BroadcastId, Previous - Address >$

However, it is possible that a node is no longer active when a request is being reversely routed. In this case, the protocol uses Ad-hoc On-Demand Distance Vector protocol (AODV) to route the reply back to the origin from the point of failure.

The configuration module defines the variables that may optimize the behaviour of service discovery according to the environment. Namely, the advertisement frequency, number of hops it should expand, the lifetime of an advertise, the number of service cache entries and the lifetime a reverse route entry should remain in cache.

3.4 Location System Components

3.4.1 Spotter

The Spotters are the leaf nodes of the location system. They are responsible for all the sensing that occurs in the system. This means that these nodes differ from the rest in the fact that they must have access to the hardware that enables them to sense people or related information. The spotter architecture can be seen in Figure 3.6.

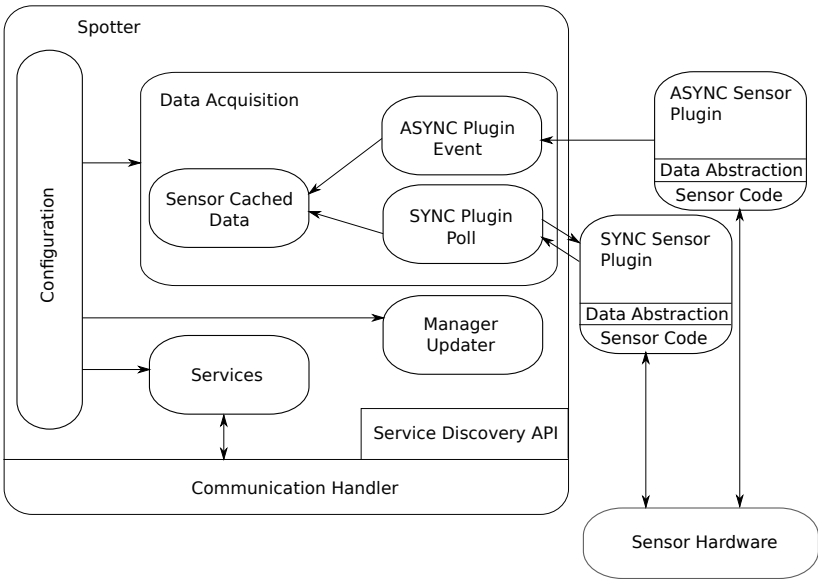


Figure 3.6: Spotter architecture

One of the requisites of this system was to integrate multiple and new technologies for sensing people. For that reason, we designed a plugin based architecture. Therefore, for each type of sensor existent in the system a plugin has to be written so that the spotter can handle the technology. These plugins implement the protocol with the sensor hardware and process it's information so that it can filter noise data, and create an abstraction of the data in such a way that all nodes can understand. Plugins can be of two types: synchronous or asynchronous. Synchronous plugins only activate their sensing mechanisms when prompted by the spotter. On the other hand, asynchronous plugins react to events triggered by people's movement or appearance, for instance, when an infrared beam located in an elevator's door gets interrupted, it usually means that a person just walked in or out of the elevator, therefore an asynchronous event occurs. As for the abstraction of the sensor data, and regarding our location system it can be of four types:

- *PEOPLE COUNT* - Type of sensor data that represents the number of people present in a

certain area. For instance, infrared heat sensor plugins that process the infrared image to infer the presence of people.

- *PEOPLE ENTRY/EXIT* - Indicates an entrance or exit of a number of people to a certain area. For instance, infrared beams located at the entrances of an area can detect the entrance or exit of people.

- *PEOPLE PINPOINT* - This type of data indicates the exact location of one or more people. For instance, an RFID reader with short range can pinpoint people associated with an RFID tag to the reader's self location.

- *Radio Signal Strength (RSS)/DISTANCE TO SENSOR* - This data type indicates the radio signal strength of in range people's devices. For instance, a Bluetooth sensor can find in range Bluetooth nodes and get their signal strength. Through the signal strength the plugins can also try to achieve the distance to the sensor. This data type is used for post-processing by the Manager node and it will be discussed later.

When a plugin is done processing the information, it delivers the abstracted data to the spotter so it can later send it to the Manager (will be analyzed in the respective section) along with all other sensor data. In order to do that, the spotter must have an associated zone manager. This may occur in two manners, both using the service discovery module described in section 3.3.2. The manager can request the spotter service with a given frequency of update or the spotter can request for it's manager to the service discovery and, upon discovery, spontaneously register with the manager to start providing it's service. Either way, both services negotiate and agree the frequency in which the spotter updates the manager, taking into account the minimum time the spotter takes to acquire sensor information. Moreover, the spotter also provides a service for requesting instant sensor data available in the node. This means that different purpose systems can subscribe for information to the spotter or even get it's current information at a specific instant, which makes the spotter easily portable to other systems.

Finally, the configuration module acts upon the other modules present in the spotter. It is used so that it can adapt more accurately to the associated environment. For instance, it tells the data acquisition module what are the plugins available, what are their respective types (synchronous or asynchronous) and what is the frequency for polling the synchronous plugins. It also configures what is it's own physical position and what is the minimum period between updates to the manager.

3.4.2 Zone Manager

Zone managers are nodes that contain location information about the zone they are managing, therefore, there is only one manager per zone. The manager architecture can be seen in figure 3.7.

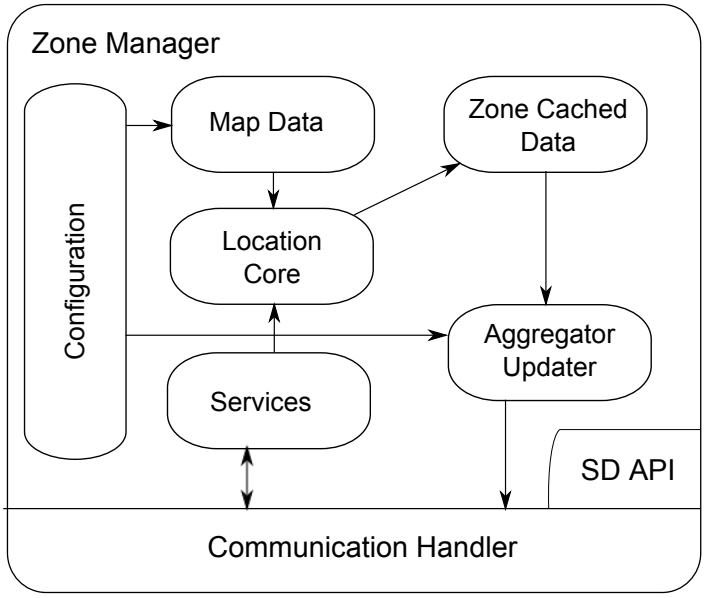


Figure 3.7: Zone Manager architecture.

In order to compute the location of people, the manager must have at least one spotter associated. Therefore, the manager regularly requests for all the spotters located in his area through the service discovery API. From this point it will start receiving asynchronous service replys as they are found. When a service does not belong to the set of known spotters, the manager sends a request for it's service with a desired frequency. As explained in the previous section, there can also be spotters registering spontaneously with the manager. Similarly, the manager can also spontaneously register with the aggregator node if it does not have one associated. In order to do that, it also requests the service discovery API for an available aggregator. Upon discovery, they agree with the frequency in which the manager updates the aggregator and from that moment on the manager periodically sends the state of the area, namely, the number of people in the area, and, if available, their physical location.

Much like the spotter, the manager also provides the instant state service. This means that other services can request for the state of a certain area at a certain time. It also provides a service for spotters to spontaneously register and another to deliver their sensing data. The aggregator or other types of nodes can also request the Manager for a frequent update of it's state, which can be seen

as a service subscriber.

Another important module in this application is the map data. This module characterizes the area which the manager is responsible and it is useful for aiding the location core to better calculate the location of people. A map is represented by a bi-dimensional group of hexagonal cells. The cell size can vary between maps, depending on the actual region size and desired accuracy and precision. This module also has information about the floor in which this region is located and the number of horizontal and vertical cells. Furthermore, each cell indicates if there is an obstacle present (and therefore impossible for a person to be), or if it is a transition cell. Transition cells are a special kind of cell that indicates a possible transition to another region, and specifies to what region and destination cell that transition is made. These cells are usually located at the edges of a map which are associated with doors or hallways but they can also exist in the middle of a map, for instance stairs in the middle of a division, and therefore a transition to a different floor is possible. Although these transition informations are present in the manager, this node does not make much use of it as it will be more suitable for the aggregator.

The main function of the configuration module in this application is to populate the map data based on it's current area and to provide the aggregator updater it's minimum update interval.

The Location Core module is where all computations of locations occur. When a spotter delivers data to the corresponding service, this delivers it to the location core. The behavior of this module depends on the number of associated spotters as long as the type of information that is receiving. It is important to notice that the use of the platform synchronization module is used in this context to detect data out of order and relative to older instants. As it was mentioned in section 3.4.1, there are four types of sensor data, we will now present the behavior of the module for each type.

- *PEOPLE COUNT* - For this kind of data the module assumes that the spotter from which is receiving the data is capable of counting the people in the whole area, and that it should only exist one spotter with this type of sensing per area. Therefore, it changes directly the state of the number of people in the area with the received value. If it is impossible for one spotter to count the whole area one of two solutions should be used: divide the area in two or more areas so that it can count the whole area, or maintain the state in the spotter of how many it can see and transform these values to entrances or exits of people.
- *PEOPLE ENTRY/EXIT* - Inside any area there may be only one between the two types of sensors: PEOPLE COUNT or PEOPLE ENTRY/EXIT. As this type of data is received it is straightly added or subtracted to the counter of people in area. If this counter becomes less than zero and the manager has still information to receive from other spotters, it is most likely

that it has still entrys to receive since it is possible for an area to have more than one entrance or transition cell.

- *PEOPLE PINPOINT* - As it was explained earlier this type of data is used when a spotter is sure of a physical location of people. Therefore, by receiving this information, the spotter straight forwardly adds a representation of the people to the corresponding locations. There may be several nodes of this type in an area.

- *RSS/DISTANCE TO SENSOR* - This type of data is the one that requires the most computation from the node. The behavior of the module when receiving this type of data depends on the number of spotters that supply this kind of information. In the case that only one spotter is supplying this information then the manager assumes a symbolic location such that it only considers that the person is situated in his own area. If instead, it receives more than one data of this type, it will try to calculate each person's physical position inside the area through triangulation mechanisms. In this case the mechanism used is lateration since that knowing the location of three points to any reference and their distance to another point it is possible to calculate the exact position of that point. However, when the manager only receives information of a certain person from two spotters we assume that by not being in range of a third node, it will be further away from the maximum range of the technology and, therefore, we use the coordinates of the spotter closer to the other two and with a distance corresponding to the maximum range of the technology plus the size of a cell. For each triangulation made, the result is converted to cell coordinates and the cell is checked for obstacles. If an obstacle exists in the resulting cell, the modules analyzes the neighboring cells without obstacles and chooses the most probable cell in which a person is located. For each person it discovers the location, it will add a representation of the person to the corresponding cell position. Note that this type of data can not be processed at the time of delivery since the manager needs to receive the data from all spotters who have this information to a given instant so that it can be process all locations.

3.4.3 Aggregator

The Aggregator node is responsible for maintaining the global state of localization of all areas, movements of people between areas and resolve location conflicts such as people being located in multiple areas or by more then one technology. The architecture of the aggregator can be seen in figure 3.8.

As we can observe, the aggregator has an architecture quite similar to the other nodes, especially the manager. However, some of the modules differ significantly in their functionalities.

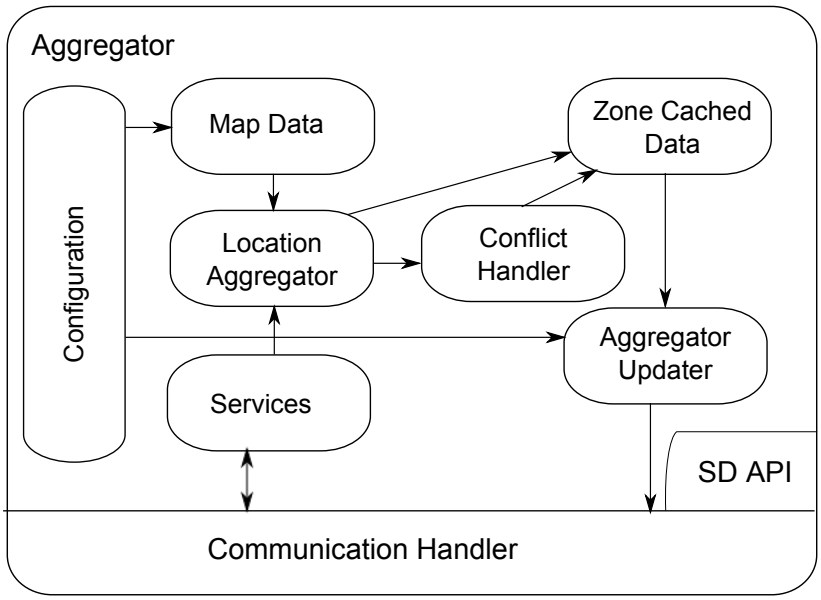


Figure 3.8: Aggregator architecture.

Maintaining the same logic of services provided by the other nodes, the aggregator also provides a subscription service, an instant state service, a spontaneous register and a service for delivery of data both from zone managers. Once again, this node uses the discovery of service to find all the zone manager nodes and subscribe them.

The UMP server updater sends, with the desired frequency, the global state of location to the UMP server or to any other service that subscribed this application. In this case, the node does not register spontaneously with any service since there is no more notion of registry with a superior entity.

The map data module is also similar to the one existent in the zone manager, however, this one contains information of all the areas that are being monitored. It also gives support to the location decider and conflict handler by supplying them transition cells, adjacent cells to a given cell, translates cell coordinates into distances and vice-versa. More description on how each map is characterized was provided in section 3.4.2.

As for the global cache, it does not only provide the updater with the current data for sending but also to improve the computation of the location aggregator and conflict handler. This cache maintains the current state of the location system, the data still to be processed and the state of the previous instant so that the other modules can better decide based on the location of people in the previous moment.

The configuration module is mainly responsible for populating the map data with all the information existent for all areas.

The location aggregator module, as the name suggests, aggregates the location information received from each zone manager and transforms into a unique source of global location and saves it into the cache. This module also checks for the existence of location conflicts and, if so, reports the conflicting locations to the conflict handler that will process the information and decide which of them is the correct location. This module is also responsible for cleaning up old information from the cache.

The conflict handler is triggered when a location conflict is detected by the location aggregator. This module receives the data relative to the previous location of the person and the new conflicting locations. Based on this information, checks whether the previous and conflicting locations have adjacent transitions. From this point, several resolutions may occur:

- In case the previous position does not have adjacent transitions and one of the new conflicting positions belongs to the same area, then the module resolves with this location.
- In case both conflicting positions have or do not have adjacent transitions to the previous area then it will calculate the distances of both to the previous location and return the position with the shortest distance.
- In case only one of them has an adjacent transition to the previous area then that position is chosen.

These statements cover all scenarios. To help illustrate this we can see the pseudo-code shown in algorithm 1.

Analyzing the algorithm we can easily associate the first sentence to the first block of code and the remaining code is common to whether the previous location has or has not adjacent transitions. This algorithm favors the remaining in the same area since it is acceptable that when a person is starting to reach a transition cell another area may capture it's signal too. The next reasonable event is when the previous location has adjacent transitions to more than one area, and both those areas claim that this person is there, in this case the algorithm tries to guess the right position by calculating the closest position to the previous. Lastly if only one of those areas have transitions to the previous area, it is probable that the other area is on a different floor and so, the algorithm chooses the first position.

Algorithm 1 Resolve location conflicts

if not ($previousLocation$ has $AdjacentTransitions$) **then**
  if $previousLocation.area$ = $newLocation.area$ **then**
    return $newLocation$
  end if
  if $previousLocation.area$ = $conflictingLocation.area$ **then**
    return $conflictingLocation$
  end if
end if
if ($newLocation$ **and** $conflictingLocation$ have $AdjacentTransitions$ **to** $previousLocation.area$) **or** ($newLocation$ **and** $conflictingLocation$ have **not** ($AdjacentTransitions$ **to** $previousLocation.area$)) **then**
  return $ClosestPosition(newLocation, conflictingLocation)$
end if
if $newLocation$ has $AdjacentTransition$ **to** $previousLocation.area$ **then**
  return $newLocation$
end if
if $conflictingLocation$ has $AdjacentTransition$ **to** $previousLocation.area$ **then**
  return $conflictingLocation$
end if

3.4.4 UMP Server

The UMP server is the system that performs the computation of patterns of movement. This node is situated in the local network and has superior processing capabilities, memory and storage resources when in comparison with the nodes that contain the other components. The architecture of this system can be seen in figure 3.9.

Just like the other components, this application also uses the multi-functional platform and as such, can also use the discovery of services module even though it is located in a distinct network. Therefore, the UMP server uses the corresponding Application Programming Interface (API) to discover the aggregator node of the desired area and subscribe its service. There may be several aggregators on the network, and this server, can even subscribe to more than one. However, following the ideal topology created for the system, the data provided by each aggregator could not be correlated since they would belong to distinct environments.

The map data module is similar to the module in the aggregator. It has information of all areas in the respective environment and provides different data to the modules. For one hand, it provides the graphical interface the tools to represent the areas to the user and to place the people in the according cells. On the other hand, helps the statistical module to analyze possible intermediate cells when it receives movement between non-adjacent cells.

This application when subscribed to an aggregator, periodically receives the global state of location. This information is passed to the statistical module that processes this information to obtain the

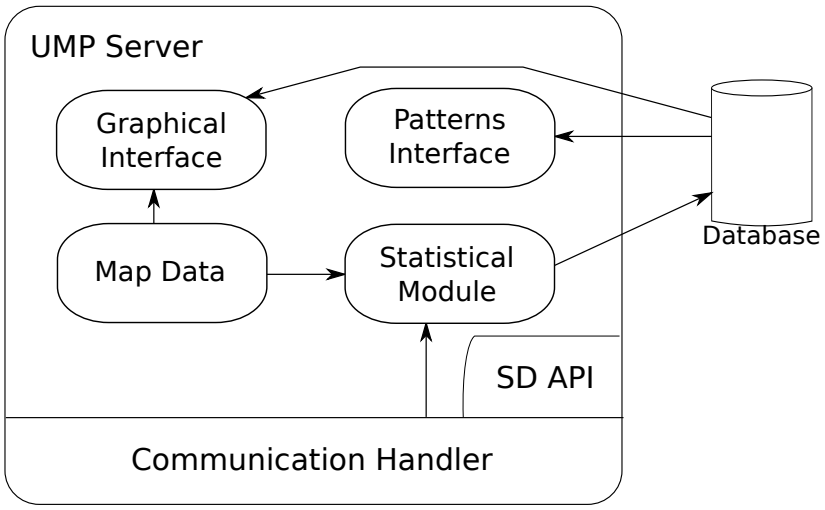


Figure 3.9: UMP Server architecture.

movement between cells from the previous instant to the current position and conveniently writes to a database in a statistical manner, such as day and time of movement, origin of the initial movement, among others.

The graphical interface module, as the name suggests, is the module that provides an user interface to visualize diverse statistical data concerning the patterns of movement, giving the possibility to visualize the last location state received from the aggregator and define specific time intervals and areas to calculate the respective patterns.

The export interface module is the module that provides the services so that other applications can obtain the patterns of movement relative to a building or space. Therefore, the node also uses the discovery of service API to provide this service so that other nodes can easily find it. This module is useful, for instance, for applications that control domotics so that they can act on the devices in a more intelligent way. For instance, turning on and off individual air conditioning systems based on the observed patterns to a certain area.

3.5 Summary

In this chapter we described the multi-functional platform for outdoor and indoor monitoring, discussed the integration of a service discovery module and the effects on this platform and presented the architectures of each component of the location and tracking system individually and as a whole.

Chapter 4

Implementation

4.1 Introduction

This chapter will analyse our decisions in terms of implementation of the system prototype and also on the specific implementation and installation of use case system in IST-Taguspark campus. We start by describing the process and environment where the development and installation were sustained, followed by the general method we used for communication between each component, then we take a look on how we implemented the service discovery so that it could integrate the platform, and analyze each component of the location system.

4.1.1 Development Environment

The implementation of the location system prototype as long as the multi-functional platform and integrated service discovery were developed on TS-7500 boards. These boards are adequate for this project since they provide the tools for integrating sensors, either simple or complex, are small sized, have a low cost and are capable of running Linux Operating System (OS). The board contains two USB ports and one ethernet port, and a type B USB entry that powers the board. It also has a set of Digital Input/Output (DIO) pins that are usually used to communicate with sensors either with protocols like SPI and I^2C or with simple two-state devices (HIGH and LOW). Because of the need to communicate at a low level with the board and also the restricted resources they possess, we opted to implement the prototypes in the C programming language with the boards running the Debian Linux OS.

4.2 Data-Interchange

Before discussing the implementation of each component designed by this thesis, we will describe the implementation of the data-interchange format. Some of the data transfered from the different type of services is very complex, with variable structure sizes and this can make the receiving system more complex when not using any notation format. Therefore we felt the need to use an extensible format, that was easily readable and did not jeopardize the performance nor added much overhead to the sent data, since the services are on a limited resources environment. We analyzed Extensible Markup Language (XML), Javascript Object Notation (JSON) and Yaml Ain't Markup Language (YAML). As for XML, although it is the most widely used markup language and is probably the most easily readable and extensible format, it introduces a high overhead and its generation and parsing is considerably heavy. The YAML is a very easy to read format with good performance and little overhead. JSON is a little harder to read than YAML, however it has the best performance ant lowest overhead within the analyzed formats. Therefore, we chose to use JSON for interchanging complex data between services. We used a library implemented in ANSI C for generating and parsing JSON data, the Yet Another JSON Language (YAJL)[1]. We implemented a common library that traduces all the data packets existent in the implemented services, including the service discovery, into JSON and also generates the packet from received JSON data.

4.3 Service Discovery

The service discovery module implements the GSD protocol. Nonetheless, it is integrated into the multi-functional platform and, as such, has modified structures. As described in section 3.3.2, this module is just like an application that functions as an add-on to the platform. What makes this application different from the others is the fact that it provides an API so that other applications can use it. This API implements the respective local requests to the module through the communication handler of each application. However, for simplicity purposes the service discovery module instead of receiving the registers of local services in runtime, is registering the local services upon start by parsing a configuration file. Relatively to the GSD protocol, the service match is slightly different since the services are described differently. Since Web ontology Language (OWL) is complex and even though it allows great flexibility, for our specific scenario we did not needed the potential of such language and, therefore, implemented a simpler one dimension array group hierarchical representation with a possible description attribute. For instance, a spotter located on zone A could be described as SPOTTER:A and belonging to the groups ZONE_A $\rightarrow$ SPOTTER $\rightarrow$ LOCATION_SYSTEM $\rightarrow$ SERVICE.

[1] `http://lloyd.github.com/yajl/`

Upon start, the configuration module receives three configuration files as arguments. A handler configuration file, a gsd configuration file and a services configuration file. The handler configuration file is the file where is defined the type of transport to use (reliable or unreliable), the addresses of the handler interface and broadcast address if needed. The configuration module just has to pass this file as argument upon creation of the handler. An example handler configuration file is present in figure 4.1.

```
<handler id="1001">
 <transport type="unreliable"/>
 <address type="inet" addr="172.20.41.116" port="32121" enable_broadcast="true"/>
 <address type="unix" addr="/tmp/1001"/>
</handler>
```

Figure 4.1: Example handler configuration file.

The gsd configuration file is where the main constants of the protocol are tweaked to better adapt to the current environment. This file has a basic variable equals value format. It defines the time interval in seconds between advertises, the lifetime in seconds an advertise remains in cache, the diameter (number of hops) an advertise reaches, the maximum number of advertises the cache stores and the timeout in seconds in which a reverse route entry is kept in the table. It also defines some auxiliary constants like the handler address of the module, what port is it binded to, the broadcast handler id (to emit in broadcast) and the interface it is using for communications (to get its own ip address). An example gsd configuration file is represented in figure 4.2.

```
ADV_TIME_INTERVAL=25
ADV_LIFE_TIME=255
ADV_MAX_DIAMETER=1
ADV_CACHE_SIZE=30
REV_ROUTE_TIMEOUT=50
MYADDRESS=1001
MYPORT=32121
BROADCAST_ID=1000
INTERFACE=wlan0
```

Figure 4.2: Example GSD configuration file.

The last configuration file is the file where the services available in the node are described to be registered in the module. This file is a bit more complex because involves an undetermined number of services and groups. Therefore, this file is passed in JSON format. For each service is indicated its description, handler address, port and groups. An example configuration file is present in figure 4.3.

This file always starts with the key *Services* which opens an array. Each entry of the array is an object that represents a service. Then for each property of the Service we define a key-value pair in which the *description* is a string, the *port* and handler *address* are numbers and the *groups* is

```
{"Services":
  [
    {
      "description":"SENSOR:2",
      "port": 32122,
      "address": 1002,
      "groups": [ "ZONE_2", "SENSOR", "PEOPLE_LOCATION", "SERVICE"]
    },
    {
      "description": "MANAGER:2",
      "port": 32123,
      "address": 1003,
      "groups": [ "ZONE_2", "MANAGER", "PEOPLE_LOCATION", "SERVICE"]
     }
  ]
}
```

Figure 4.3: Example provided services configuration file.

another array with the corresponding hierarchy group strings. Thus, in this case, this node would be providing the spotter and manager services, both from zone 2.

4.4 Location System

The implementation of the location system may be divided into two categories: the generic platform of the location system and the specific use-case prototype for installation in IST-Taguspark. The platform was implemented taking into account the logic architecture described in section 3.2. In all the different components that are part of the system there are common structures they use. The communications handler provided by the multi-functional monitoring platform is one of them. Upon initiation, these applications register the handler so they can use the platform and be accessible by other services. In every outgoing communication wanted, the handler is used to send the data to another application either in the same node or external node. Furthermore, the GSD Application Programming Interface (API) is used by every application to use the discovery service module to find, in this case, their corresponding logical parent. As mentioned before, all applications also use a common library to send data in JSON format and translate JSON data into a recognized data structure. Finally, there is a common map library that contains operations over map data like the loading of maps and the translation between physical coordinates to cell coordinates and vice-versa. The specific implementation of each component will be discussed individually.

Considering the installation in IST-Taguspark we have built a laboratory prototype in which we monitored two areas: One only counting entrances and exits, and the other also doing pinpoint of physical location of people within the area. In order to choose the most adequate technologies to use we considered the following requirements: cost - the cost of possible devices should be low and

preferably using already present technologies to lower infrastructure costs, also, the computation needed should also be low as nodes have scarce computation capabilities; acceptability - acceptance from people to be localized using one technology (usually concerning privacy issues); accuracy and precision - the technology should provide localization accuracy of up to 20 meters; intrusion - people should not need to use specific devices or install specific applications in their mobile devices (remote-positioning); scalability - it should allow the scaling to large buildings and campus without greatly increasing complexity and difficulty; technology limitations - it should work indoors. Regarding people counting we can conclude that the video-based counting has associated high costs of infrastructure and computation capabilities; the Radio Frequency Identification (RFID) technology fails on the acceptability of people to wear RFID tags and their distrust concerning privacy. Although infra-red technology provides less accuracy in people counting than the previous technologies, it still provides an acceptable accuracy and it has a better evaluation on all the other properties. As for people localization we deduce the following: using Global System for Mobile Communications (GSM) technology is intrusive (self-measuring) and it does not provide good enough accuracy; infrared and video-based localization have high computational needs; the use of RFID in localization besides the acceptance issue, it has limited scalability and high infrastructure costs; the Wi-Fi and bluetooth technology are both a good fit for localization considering the previous requirements. Therefore, we chose a solution using the infrared technology for people counting and bluetooth technology for localization.

4.4.1 Spotter

This component implements all the features described in section 3.4.1. The spotter is run with the handler file configuration and with the spotter file configuration. The first is similar to the one seen in 4.1, applied to the spotter data. The later lets the configuration module know what the interval in seconds in which synchronous plugins should be queried, the minimum interval in seconds to update the manager, the area id and cell coordinates in which the spotter is located, its handler address and the plugins available. An example configuration file is presented in figure 4.4.

```
SENSE_INTERVAL=15
MIN_UPDATE_INTERVAL=20
MY_ADDRESS=1002
MAP=2
LOCATION=1;0
PLUGIN=plugins/ir_sensor.so ASYNC
PLUGIN=plugins/bt_sensor.so SYNC
```

Figure 4.4: Example spotter configuration file.

Note that for each plugin, it must be indicated what the path to the plugin compiled file and followed by the type of plugin linking (synchronous or asynchronous).

The services module, for each packet received and after generating the packet from JSON data, checks for what the packet type and, depending on it, executes the corresponding service. As for the data acquisition, there are several methods for implementing plugins. Our implementation is based on dynamically linking the plugins in runtime. The spotter interacts with the plugins through well-defined function signatures that they must implement:

void start_plugin(void (* sensor_result)(SensorData *)) This function initializes the plugin. It receives a pointer to a callback function that receives the resulting sensor data. This should be stored in the plugin and called when there is available data.

void stop_plugin() This should implement the cleansing of structures and termination. After this function is called the plugin is usually unlinked from the spotter.

void start_sense() This function is only used for synchronous plugins. The spotter periodically (SENSE_INTERVAL) calls this function on every synchronous plugin available that should implement the actual sensing of the environment to later provide the results.

The configuration module lets the spotter know which are the plugins to be used, those are dinamically linked and for each one the spotter loads the *start_plugin* symbol (returns a pointer to the method) and calls it. At this point all plugins are initialized and contain the pointer to the callback function in the spotter that receives the sensor data. When in this state, the asynchronous plugins are already sensing the environment/ready to be triggered by an event and report it to the spotter. On the other hand, the synchronous plugins are not active and only start sensing when probed by the spotter. When the sensing is finished, executes the callback with the resulting data.

For the application scenario of the location system in IST-Taguspark we decided to implement two technology plugins:

- An infrared beam sensor plugin that detects the entrance/exit of people in a division in an asynchronous manner.
- A bluetooth sensor plugin that detects the radio signal strength of in range devices.

Infra-red

We used two types of infrared sensors that operate in distinct ways. The first type of infrared sensor (can be seen in figure 4.5) is a short range sensor (approximately 80cm) that is simultaneously

emitter and receiver. The sensor is constantly emitting an infra-red beam, and when in open space it does not receive any signal. Whenever the sensor is in this is state its output is HIGH. When the beam is obstructed at a short distance it reflects on the person or object and is received by the sensor. In this case the sensor changes its output to LOW. Thus, this output is directly connected to a DIO Header in the TS-7500 board that is capable to read this signal. This sensor is ideally placed on the sides of single passage doors since the sensor range is sufficient to detect any passage. In order to detect the direction of movement of a person (entrance or exit) we use two of these sensors placed in parallel and each one connects to an individual DIO. These sensors are powered directly by the board with 5 volt power source. The prototype is displayed in figure 4.6.

Figure 4.5: Short range infrared sensors.

The other type of infrared sensors have a range of tens of meters and require a pair of emitter and receiver (figure 4.7). Thus, its operation is inverse to the prior sensor. In this case, when the receiver is receiving the beam, it means that there are no obstructions. In case it is not receiving, then the beam was obstructed by a person or object. Furthermore, we faced some issues when using the same method as above for detecting movement direction. This was due to the fact that the emitter has such an emitting power and the restriction of the beams being closer then the width of a person in profile (would lead to a misinterpretation of people standing between the beams), that when a beam was obstructed, both receivers still got the signal from the unobstructed beam. The solution was to resort to sensors that could operate in synchronous mode, that is, that one pair is never operating at the same time as the other and alternate at a high frequency. This is achieved by feeding the sensors with alternate current in which each pair is in inverted polarities. Thus, we use 220 to 12 volt transformers with alternate current. The physical prototype can be seen in figure 4.8. These receivers contain relays powered by the board with 3.3 volts that are switched when the

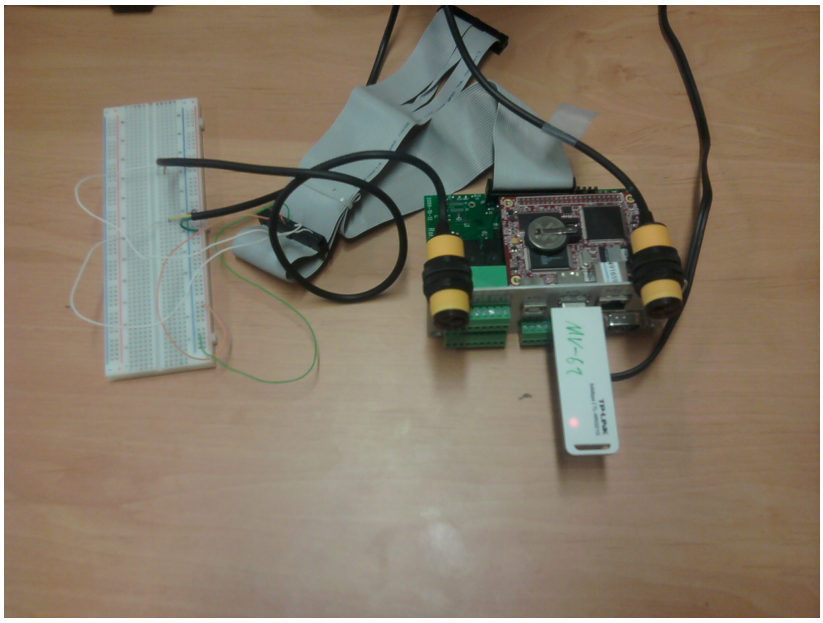


Figure 4.6: Prototype of a node with short range infrared sensor capabilities.

beam is obstructed. The output of this relay is then directly connected to DIO headers in the board so that it can interpret the state of the sensor.

Figure 4.7: Infrared long range emitters.

The plugin structure is the same for both cases, the difference is in the expected states which are opposed to each other (HIGH and LOW). In algorithm 2 we can see the pseudo-code referent to the first type of sensor.

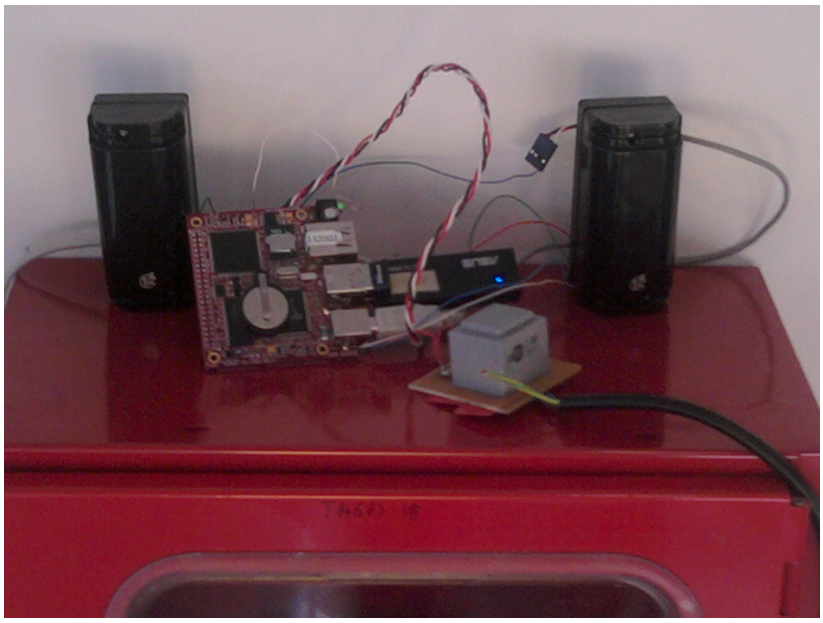


Figure 4.8: Prototype of a node with long range infrared sensor capabilities.

Bluetooth

The Bluetooth sensor is an USB device that provides the technology functionalities to the system it is connected. In order to be used, we enabled the bluetooth support on the ts-7500 installed operating system. The bluetooth version of the device is 2.1 and the its range is approximately 10 meters. We also used the BlueZ library that implements the bluetooth protocol.

The bluetooth plugin is a synchronous plugin that, upon request, enters the inquiry mode to discover all in range nodes. An inquisitions takes about ten seconds. During this period it is possible to check the signal strength of each discovered node. Since the signal strength is fairly inconstant and is subject to several factors that can increase the attenuation, during the period of inquisition it is calculated the average signal strength for each node. This value is then used to calculate the corresponding distance. This distance can be achieved with two methods, depending on the plugin configuration:

Open space method This method calculates the distance to the node based on the propagation and attenuation laws of radio signals in open space. This is ideal for outdoor areas like gardens, clearings and others. The attenuation is related with the emitting power over the square of the distance. Therefore the corresponding formula in decibels resolved to the distance is:

$$PATHLOSS = -27.56 + 20log(d(m)) + 20log(f(MHz))$$
$$\leftrightarrow d(m) = 10^{\frac{PATHLOSS-40.05}{20}}$$

Algorithm 2 Infra-red sensor plugin

```
in ← HIGH
last_in ← LOW
out ← HIGH
last_out ← LOW
loop
  if (in = LOW and last_in = HIGH) then
    in_time ← time
    if ((in_time − out_time) ≤ DELTA_MOVEMENT) then
      sensor_result(1)
      in_time ← 0
      out_time ← 0
    end if
  end if
  if (out = LOW and last_out = HIGH) then
    out_time ← time
    if ((out_time − in_time) ≤ DELTA_MOVEMENT) then
      sensor_result(−1)
      in_time ← 0
      out_time ← 0
    end if
  end if
  last_in ← in
  last_out ← out
  sleep(SENSE_INTERVAL)
end loop
```

Fingerprinting method This method is based on the pre-observation of the behavior of signal strength in account for the distance to the node. This is best for indoors, since the radio signals are substantially affected by the obstructions, reflections and multi-path effects. In this case, the plugin receives a configuration of the signal strength intervals that correspond to a determined distance step. An example configuration could be:

50cm: -30dB to -40dB
100cm: -41dB to -50dB
150cm: -51dB to -57dB

In this example, the result of each inquisition would be distances of 50cm steps. This is a trade-of between accuracy and precision. If we shorten the steps we could increase the accuracy but may jeopardize the precision. On the other hand, if we increase the steps we will be lowering the accuracy but increasing for certain the precision. However, we can also configure the plugin with nonuniform steps if we have more certainty of the possible distances of people to the spotter. This method may increase both accuracy and precision but requires more study to proper analyse the area and the signal strength. For instance:

37cm: -30dB to -44dB
86cm: -45dB to -60dB
150cm: -61dB to -87dB

4.4.2 Manager

The manager implements all the modules described in section 3.4.2. Just like the spotter, the manager's configuration module loads the handler file to register the handler and define the minimum interval in seconds to update the aggregator, its handler address and the map file to load. This map file contains all the map data information that will be loaded into memory. This file is in JSON format and may become complex of configuring depending on the size of the area, desired accuracy and quantity of obstacles and transitions. One of the areas where the system was installed was the one represented in figure 4.11. The red color indicates the entrance/exit of the room, the black colors indicate obstacles like tables and cabinets, the white space represents free area.

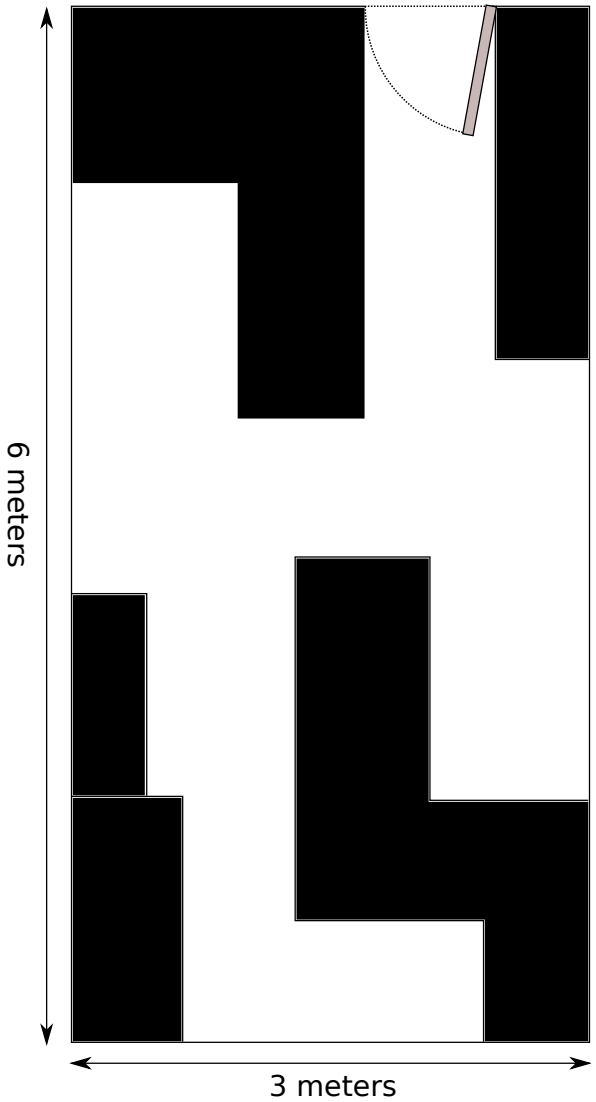


Figure 4.9: Example physical location area.

Based on the plant of the room we can do a close enough map approach using 75cm cell size. In figure 4.10 is the representation of the room using hexagonal map cells. The red cell represents

a transition cell.

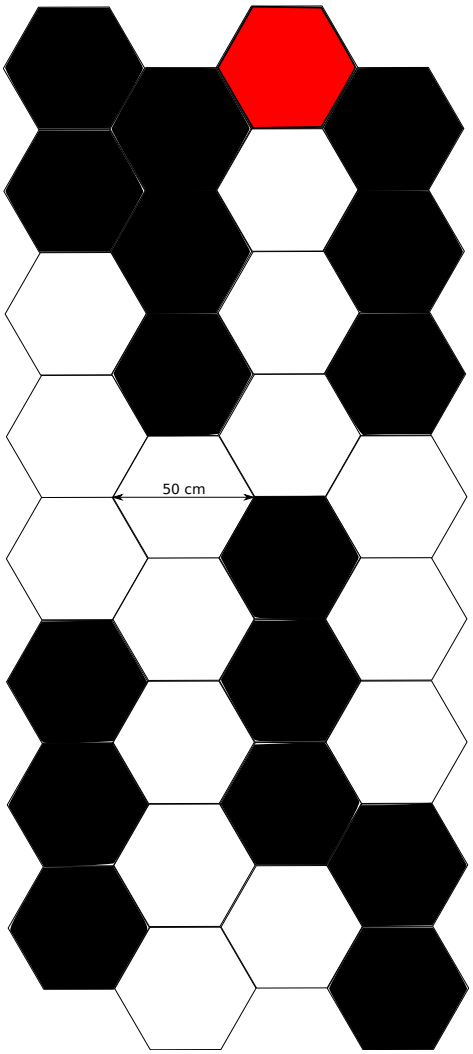


Figure 4.10: Cell representation of the example location area.

After this step we already have enough information to describe the map accordingly. Thus, the resulting configuration file would be as follows:

The *map_id* property is an unique identifier of a map in a location system, in this case we identified it as 2. The *cell_size* is defined in centimeters and according to the map figure we defined it with 75. The *x_cells* and *y_cells* are respectively the number of lines and columns of this map. The *cells* key is the definition of each cell state. Therefore, this is a two dimensional array where the number of rows is *x_cells* and *y_cells* the number of elements. Each element may have four different values: 0 - obstructed, 1 - free, null - transition cell to unmonitored area, or an object representing a transition. In this case we have a transition to the cell (3,7) of the map with the id 1.

```
{
  "map_id" : 2,
  "cell_size": 50,
  "x_cells":8,
  "y_cells":4,
  "cells":
   [
     [0,0,{"area_id":3,
           "x_cell":0,
           "y_cell":3} ,0],
     [0,0,1,0],
     [1,0,1,0],
     [1,1,1,1],
     [0,1,0,1],
     [0,1,0,1],
     [0,1,0,0],
     [0,1,1,0]
   ]
}
```

Figure 4.11: Example map configuration file

4.4.3 Aggregator

The aggregator node implements the majority of functionalities described in section 3.4.3. This node aggregates the information from all the managers, tries to resolve location conflicts and exports the results to a specific format. The Configuration module is similar to the manager, but in this case, the loading of map data is not done to a single file but several (and probably all). Thus, the implementation of this component is based on the iteration of a given folder, to read and load all corresponding maps into memory. The other configurable property is the minimal update frequency in which updates the UMP server or exports data. In this case, there is no need to define the location or area in which this component is located since it is unique and does not influence its behaviour.

The Aggregator updater was not implemented accordingly to the architecture since the UMP Server was not yet been implemented. Instead, the aggregator exports the current state of global location data into files so they can be post-processed. The aggregator exports data into two different files: a people counter file and a people location file. The prior is mainly organized by time per area and the later by people location over time. These files can be processed for instance to generate statistical graphics, analyse movement patterns, or to create an animated representation of the movements over time.

4.4.4 UMP Server

In order to simplify the implementation of the whole project and give emphasis to the other components, we did not implement the UMP Server during the period of this thesis. However, it is possible to obtain the patterns of movement by post-processing the export files provided by the aggregator. With these files we can infer information like the entries and exit of people to a number of areas throughout the day, the percentage of people who walk the same path from one area to another, and others.

4.5 Summary

In this chapter we described the key aspects of our implementation, either in the Service Discovery and Location System components as well as our chosen approach to install a use case system in the IST-Taguspark campus. We also mentioned what was left for future development, from the original system architecture.

Chapter 5

Evaluation

5.1 Introduction

5.1.1 Objectives

The objectives of these tests were to assess the behaviour of the system when applied to a real-world installation. We expect that all nodes eventually become aware of all the others and what services they provide, that they automatically connect upon discovery of a needed service and start exchanging location information. We also expect to analyse the number of people within two areas throughout the day and to track people's movements using the bluetooth technology.

5.1.2 Environment

As mentioned before, one of the goals of this thesis was to have a functioning prototype in the IST-Taguspark campus. Due to our limited resources and funding we had to install a small use case prototype on our installations. We created and monitored two zones. The first area contains a group of division work places. The second zone is one of these divisions. The physical topology of the divisions and the location of the installed nodes are represented in figure 5.1.

In the first area, we intend to get the actual counting of people within the area by using the infra-red sensors to detect entrances and exits and also be able to detect the position of a portion of these people through the bluetooth plugin. In the second area we just intend to count the number of people present in the division. These will also count as being in the outer area. Therefore, we installed the infra-red sensor emitter and receiver pairs at the lobby of the area to detect the people's entrances. As we can see in figure 5.2 the emitters were put at the left side of the entrance and receivers at the right side. This was because the left side of the lobby gets straight sunlight by the afternoon that could lower the effectiveness of the receivers. This node (A) also contains a bluetooth sensor and runs the spotter application.

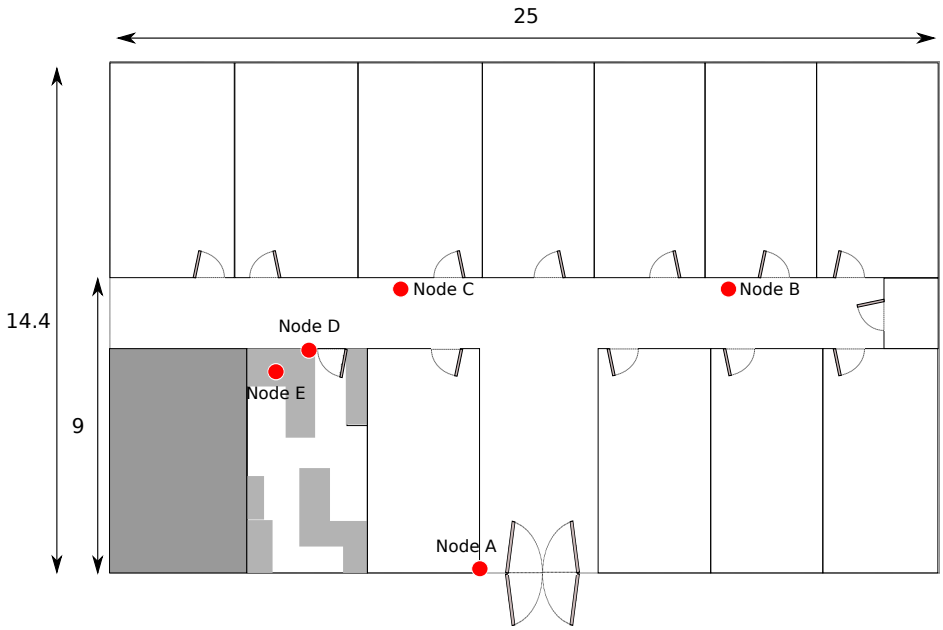


Figure 5.1: Global topology of the monitored areas and nodes physical location.

Figure 5.2: Installation of the infra-red sensors at the entrance of the outer zone.

The other two elements (nodes B and C) present in the area only have bluetooth sensors in order to compute the triangulation 5.3. They also run the spotter application.

Inside the second area, there are two nodes installed. One of the nodes (node D) has an infra-red sensor to detect the passage of people and runs the spotter application and also contains the correspondent manager of the area. As we can see in figure 5.4, these sensors were placed closer to the space of passage due to the short range of the sensors.

The last node (node E) does not do any sensing, but on the other hand is both the manager

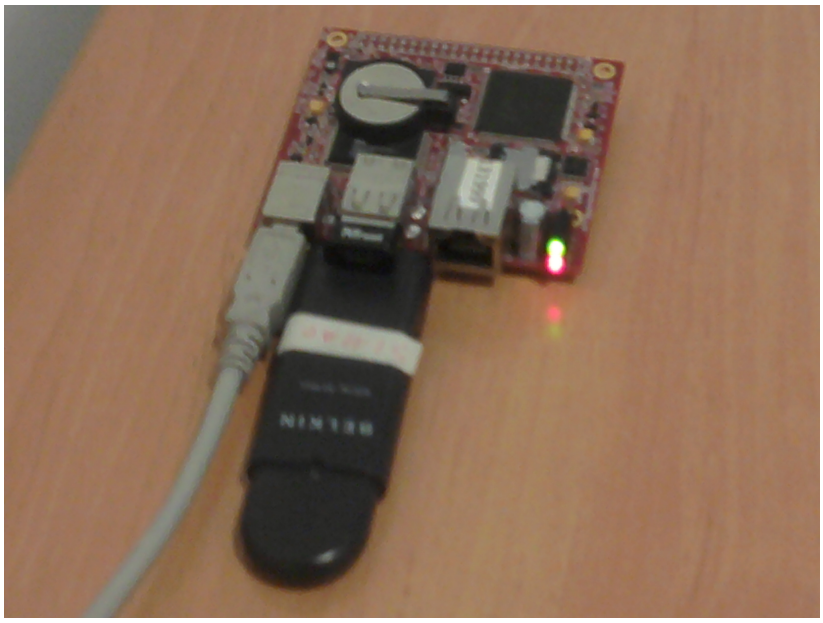


Figure 5.3: Node containing a bluetooth sensor and wi-fi USB dongle for communication.

Figure 5.4: Node at the entrance of the inner division. Contains infra-red sensor and wi-fi USB dongle.

of the outer area and the aggregator. This was conceived this way so that we can demonstrate that zone managers can coordinate nodes even though they are not on the same area. Note that all nodes also run the multi-functional platform as long as the implemented service discovery. In terms of connection, and because the location system is network independent, the nodes could be connected to each other through several technologies. However, in this installation, since the nodes were not far apart we opted to link them to a regular 802.11 access point.

Table 5.1: Functional tests to the infrared sensor plugin

Test	**Result**
Normal entry/exit	One entrance/exit
Fast entry/exit (running)	May or may not detect the entrance. As the plugin reads the sensor state with 100 Hz frequency, the sensor may be at the obstructed state for too short time and therefore, not detect the entry.
Cross entrance	In this case, the behaviour of the plugin depends on the exact moment the first beam gets obstructed. For instance, if the first beam to be obstructed is the inside beam, the plugin will count an exit. Otherwise, will count an entrance. Usually, it does not also detect the opposite direction because the beams stay obstructed until both have completely crossed the entrance.
Simultaneous entrance	If an entrance is made simultaneously in such a way that the beams do not leave the obstructed state between the two entrances, only one is detected.

5.2 Functional Tests

5.2.1 Service Discovery

Regarding service discovery we verified that, in fact, all nodes became rapidly aware of each other as expected. However, this network topology made the discovery of services fairly simple because all nodes are capable of receiving advertises made from one node. Therefore, we intend to do other experiments with different network topologies like ad-hoc networks in which there are multiple hops to reach a certain node, and insert various issues.

5.2.2 Sensor Data Acquisition

Relatively to sensor data acquisition we tested the infrared plugins by doing a series of functional tests represented in tabular 5.1.

In fact, the infrared beam sensor is not the most robust method for counting people, however, it gives a fairly good approximation for most cases. For the reasons stated in table 5.1, the most reliable method for these sensors is to put them in single passage doors in order to guarantee that a single passage is made at a time.

In the case of the bluetooth plugin, the main drawback we have faced is the fact that, in these days, not a lot of people carry their devices with the bluetooth turned on and discoverable and so we could not get a lot of results. In either case, with this plugin and especially using the fingerprinting method, this technology provides an accuracy of a few meters. Thus, for the purpose of computing patterns, getting the results for a small percentage of the moving people may be enough.

Another parameter we were trying to assess was the easiness of integrating technologies to the

spotter as a plugin. With the two plugins implemented for this thesis, we can already conclude that this parameter is subjective and is dependent on the technology. For instance the infrared plugin has a source of approximately 120 lines, however the bluetooth involves a whole lot more complexity and know-how. Nonetheless, the extra work implied to integrate a sensor in the system is fairly low in which most of it is to convert the sensor data into the abstracted form of the platform.

5.2.3 Location and Tracking System

The Location and tracking system installed has collected data approximately from 13:00 to 01:00. The next few figures show the variation of people in the areas for that corresponding time.

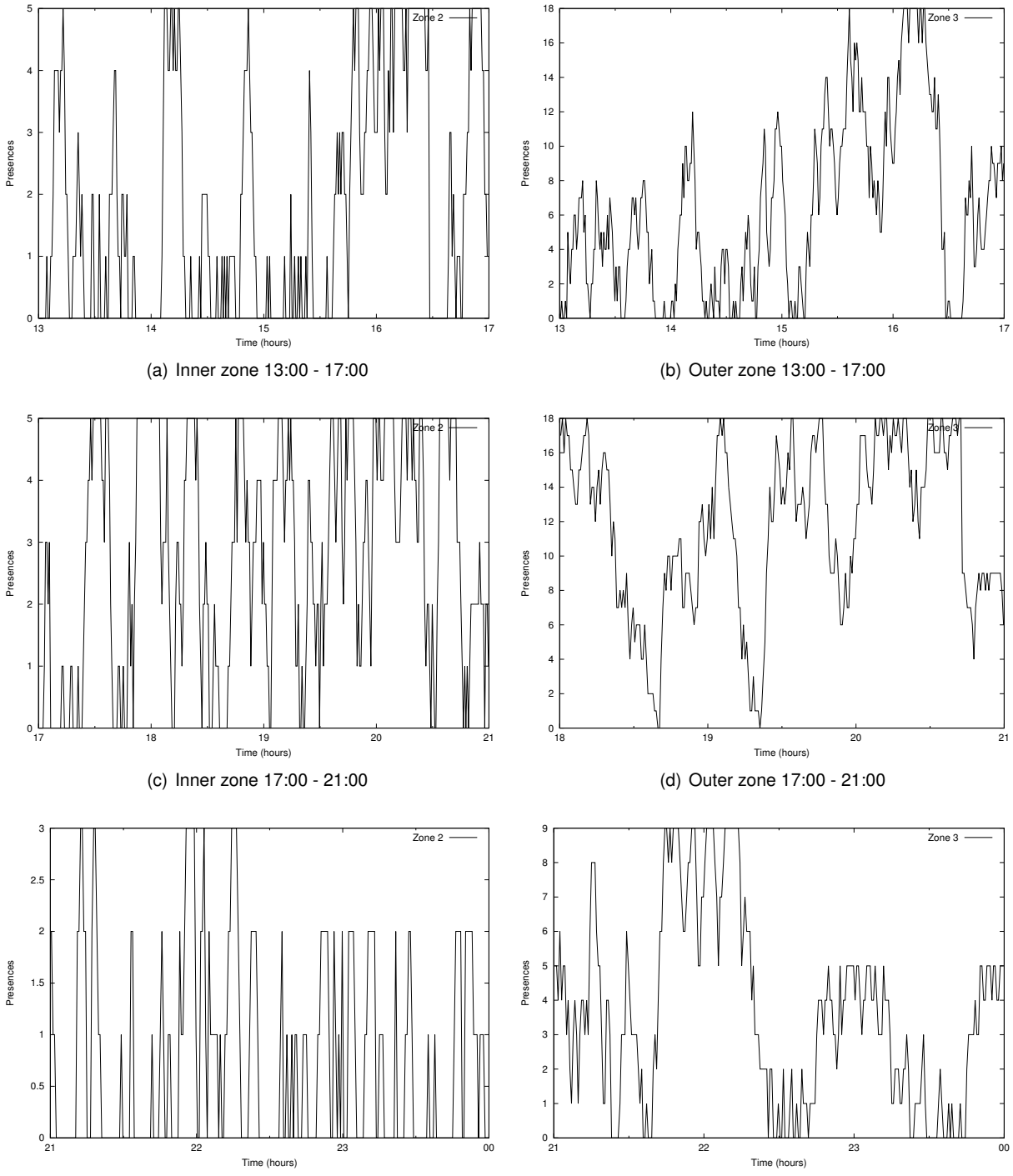
As we can see in the figures there was a lot of movement over the day. We can see that the maximum people number in the area outer area was 18 and the inner area was 5. This was expected since the inner area is a workplace for 3 or 4 people. For the reasons explained in section 5.2.2 and because of the enthusiasm and curiosity that this equipment made on people may have cause adulteration of values, such as simulating entrances of people.

Another test that we made was through the use of the bluetooth technology was the tracking of people. Unfortunately, there were only four people detected with a visible bluetooth device in this area. Two of them were certainly just passing by because they were only detected in one instant each. The third was a still pc that was detected almost through all the testing time. For this reason, it was located almost always in the same cell, and sometimes it was detected on an adjacent cell. This is because of the instability of RSS while indoors. The last person detected was our test runs, in which we have made a test paths, walking through the area and then analysed what were the results produced by the platform. In figure 5.6 is represented the movement detected by the platform and the recreation of one of our runs. Note that when the points of the detected path are slightly apart, is just for better perception because the detected movement is made by cells and so they would be exactly the same.

As we can see the results were not too far apart. We can also perceive that sometimes the movement stopped being detected. This was due to the fact that the node was no longer in range of at least two spotters. However, it was later resumed when it got back in range.

5.3 Summary

In this chapter we analysed what was the created environment to run the tests, what we were expecting, and the results provided by the platform. More evaluation is still needed to, for instance,



(a) Inner zone 13:00 - 17:00

(b) Outer zone 13:00 - 17:00

(c) Inner zone 17:00 - 21:00

(d) Outer zone 17:00 - 21:00

(e) Inner zone 21:00 - 24:00

(f) Outer zone 21:00 - 24:00

Figure 5.5: Variation of the people present in the monitored areas throughout the day.

test all types of data returned from the sensors, test the handling of conflicts and the use of multiple technologies.

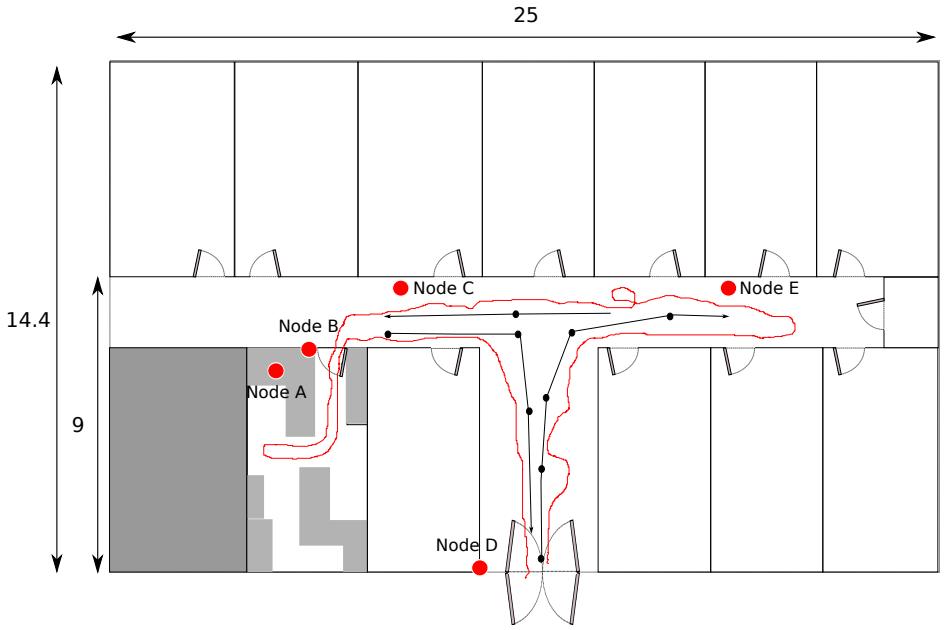


Figure 5.6: Detected movement by the platform in comparison with a real movement path

Chapter 6

Conclusions

In this thesis, we have analyzed the state of the art in the areas of service discovery presenting standard protocols and pervasive environment protocols, researched methods for generic data acquisition and low-level sensor communication protocols, referred classification properties of localization systems, the main technologies and techniques used for people counting and tracking, and means of assessing user mobility patterns. We have designed and implemented a location system that is able to integrate multiple technologies through plugins, and detect people through the corresponding sensor technologies. The localization system uses a multi-functional monitoring platform so that it can benefit of the abstraction from the communication problems. We also integrated a service discovery module into this platform. This work provide the information about the location of people and their mobility patterns so that other projects may act accordingly for reducing energy consumption, i.e., adjust the air conditioning and lighting to the real occupation of the building.

6.1 Future Work

As future work we intend to give the ability to the platform to be even more scalable by loading maps dinamically as they are needed and maintain them in memory while they are used in order to save memory and be capable of deal with very large buildings and spaces. We also intend to improve the triangulation calculation through approximations of the various distances to the node in order to always get a solvable triangulation problem. Also, provide a reset feature so that, when it was detected that, for instance, the counting of people was adulterated, one could reset the count to a certain exepected value.

Bibliography

[1] P. Bahl and V.N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784. Ieee, 2002.

[2] U. Bandara, M. Hasegawa, M. Inoue, H. Morikawa, and T. Aoyama. Design and implementation of a bluetooth signal strength based location sensing system. In *Radio and Wireless Conference, 2004 IEEE*, pages 319 – 322, 2004.

[3] M. Bertozzi, A. Broggi, A. Fascioli, A. Tibaldi, R. Chapuis, and F. Chausse. Pedestrian localization and tracking system with kalman filtering. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 584 – 589, 2004.

[4] M. Bertozzi, A. Broggi, A. Fascioli, A. Tibaldi, R. Chapuis, and F. Chausse. Pedestrian localization and tracking system with kalman filtering. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 584–589. IEEE, 2004.

[5] M. Bertozzi, A. Broggi, A. Lasagni, and M.D. Rose. Infrared stereo vision-based pedestrian detection. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 24 – 29, 2005.

[6] J. Bohn and H. Vogt. Robust probabilistic positioning based on high-level sensor-fusion and map knowledge. *month*, 2003.

[7] C. Campo, M. Munoz, J.C. Perea, A. Mann, and C. Garcia-Rubio. PDP and GSDL: a new service discovery middleware to support spontaneous interactions in pervasive systems. In *Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops. Third IEEE International Conference on*, pages 178–182. IEEE, 2005.

[8] E. Cayirci and I.F. Akyildiz. User mobility pattern scheme for location update and paging in wireless systems. *IEEE Transactions on Mobile Computing*, pages 236–247, 2002.

[9] P. Cederberg, M. Olsson, and G. Bolmsj
”o. A generic sensor interface in robot simulation and control. In *Proceedings of Scandinavian Symposium on Robotics*, volume 99, pages 221–230. Citeseer, 1999.

[10] D. Chakraborty, A. Joshi, Y. Yesha, and T. Finin. Toward distributed service discovery in pervasive computing environments. *Mobile Computing, IEEE Transactions on*, 5(2):97 – 112, 2006.

[11] Thou-Ho Chen, Tsong-Yi Chen, and Zhi-Xian Chen. An intelligent people-flow counting method for passing through a gate. In *Robotics, Automation and Mechatronics, 2006 IEEE Conference on*, pages 1 –6, 2006.

[12] S. Cheshire and M. Krochmal. DNS-Based Service Discovery. Internet-Draft draft-cheshire-dnsext-dns-sd-05, Internet Engineering Task Force, September 2008. Work in progress.

[13] S. Cheshire and M. Krochmal. Multicast DNS. Internet-Draft draft-cheshire-dnsext-multicastdns-07, Internet Engineering Task Force, September 2008. Work in progress.

[14] S.E. Czerwinski, B.Y. Zhao, T.D. Hodes, A.D. Joseph, and R.H. Katz. An architecture for a secure service discovery service. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 24–35. ACM, 1999.

[15] S. Dong. Node Placement, Routing and Localization Algorithms for Heterogeneous Wireless Sensor Networks. 2008.

[16] B. Fardi, U. Schuenert, and G. Wanielik. Shape and motion-based pedestrian detection in infrared images: a multi sensor approach. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 18 – 23, 2005.

[17] B. Fardi, U. Schuenert, and G. Wanielik. Shape and motion-based pedestrian detection in infrared images: a multi sensor approach. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 18–23. IEEE, 2005.

[18] G. Frederico. Multi-functional platform for indoor and outdoor monitoring. Master's thesis, Instituto Superior Técnico, Portugal, 2011.

[19] F.J. Gonzalez-Castano and J. Garcia-Reinoso. Bluetooth location networks. In *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, volume 1, pages 233 – 237 vol.1, 2002.

[20] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.J. Nordlund. Particle filters for positioning, navigation, and tracking. *Signal Processing, IEEE Transactions on*, 50(2):425–437, 2002.

[21] J. Hallberg, M. Nilsson, and K. Synnes. Positioning with bluetooth. In *Telecommunications, 2003. ICT 2003. 10th International Conference on*, 2003.

[22] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, 34(8):57 –66, August 2001.

[23] J. Hightower, R. Want, and G. Borriello. SpotON: An indoor 3D location sensing technology based on RF signal strength. *UW CSE 00-02-02, University of Washington, Department of Computer Science and Engineering, Seattle, WA*, 2000.

[24] J.L. Honorato, I. Spiniak, and M. Torres-Torriti. Human detection using thermopiles. In *Robotic Symposium, 2008. LARS '08. IEEE Latin American*, pages 151 –157, 2008.

[25] Jouni Ikonen and Jari Porras. Extracting and using position information in wlan networks.

[26] D. Kalinsky and R. Kalinsky. Serial Peripheral Interface. *Embedded Systems Programming*, page 55, 2002.

[27] M. Klein, B. Konig-Ries, and P. Obreiter. Service rings - a semantic overlay for service discovery in ad hoc networks. In *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*, pages 180 – 185, 2003.

[28] Michael Klein and Birgitta König-Ries. Multi-layer clusters in ad-hoc networks — an approach to service discovery. In Enrico Gregori, Ludmila Cherkasova, Gianpaolo Cugola, Fabio Panzieri, and Gian Picco, editors, *Web Engineering and Peer-to-Peer Computing*, volume 2376 of *Lecture Notes in Computer Science*, pages 187–201. Springer Berlin / Heidelberg, 2002.

[29] U.C. Kozat and L. Tassiulas. Service discovery in mobile ad hoc networks: an overall perspective on architectural choices and network layer support issues. *Ad Hoc Networks*, 2(1):23–44, 2004.

[30] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, et al. Place lab: Device positioning using radio beacons in the wild. *Pervasive Computing*, pages 116–133, 2005.

[31] L. Lee, R. Romano, and G. Stein. Monitoring activities from multiple video streams: establishing a common coordinate frame. *Transactions on Pattern Analysis and Machine Intelligence*, 22(8):758–767, 2000.

[32] D.T. Linzmeier, D. Vogt, and P. Prasanna. Probabilistic signal interpretation methods for a thermopile pedestrian detection system. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 12 – 17, 2005.

[33] S. Motegi, K. Yoshihara, and H. Horiuchi. Service discovery for wireless ad hoc networks. In *Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on*, volume 1, pages 232 – 236 vol.1, 2002.

[34] L.M. Ni, Y. Liu, Y.C. Lau, and A.P. Patil. LANDMARC: indoor location sensing using active RFID. *Wireless Networks*, 10(6):701–710, 2004.

[35] L.M. Ni, Yunhao Liu, Yiu Cho Lau, and A.P. Patil. Landmarc: indoor location sensing using active rfid. In *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, pages 407 – 415, 2003.

[36] Ling Pei, Ruizhi Chen, Jingbin Liu, T. Tenhunen, H. Kuusniemi, and Yuwei Chen. Inquiry-based bluetooth indoor positioning via rssi probability distributions. In *Advances in Satellite and Space Communications (SPACOMM), 2010 Second International Conference on*, pages 151 –156, 2010.

[37] Olga Ratsimor, Dipanjan Chakraborty, Anupam Joshi, and Timothy Finin. Allia: alliance-based service discovery for ad-hoc environments. In *Proceedings of the 2nd international workshop on Mobile commerce*, WMC '02, pages 1–9, New York, NY, USA, 2002. ACM.

[38] F. Sailhan and V. Issarny. Scalable service discovery for MANET. In *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*, pages 235–244. IEEE, 2005.

[39] Gregor Schiele, Christian Becker, and Kurt Rothermel. Energy-efficient cluster-based service discovery for ubiquitous computing. In *Proceedings of the 11th workshop on ACM SIGOPS European workshop*, EW 11, New York, NY, USA, 2004. ACM.

[40] K. Seada and A. Helmy. Rendezvous regions: a scalable architecture for service location and data-centric storage in large-scale wireless networks. In *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, page 218, 2004.

[41] S. Sivavakeesar, O.F. Gonzalez, and G. Pavlou. Service discovery strategies in ubiquitous communication environments. *Communications Magazine, IEEE*, 44(9):106 –113, 2006.

[42] Timothy Sohn, Alex Varshavsky, Anthony Lamarca, Mike Y. Chen, Tanzeem Choudhury, Ian Smith, Sunny Consolvo, Jeffrey Hightower, William G. Griswold, and Eyal De Lara. Mobility detection using everyday gsm traces. In *in Proceedings of the Eighth International Conference on Ubiquitous Computing (Ubicomp 2006*, pages 212–224. Springer, 2006.

[43] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. Service location protocol. 1997.

[44] C.N. Ververidis and G.C. Polyzos. Service discovery for mobile ad hoc networks: a survey of issues and techniques. *Communications Surveys Tutorials, IEEE*, 10(3):30 –45, 2008.

[45] M. Vossiek, L. Wiebking, P. Gulden, J. Weighardt, and C. Hoffmann. Wireless local positioning - concepts, solutions, applications. In *Radio and Wireless Conference, 2003. RAWCON '03. Proceedings*, pages 219 – 224, 2003.

[46] J. Waldo. Jini architecture overview. *Sun Microsystems, Palo Alto, CA*, 1998.

[47] H. Wang, H. Lenz, A. Szabo, J. Bamberger, and U.D. Hanebeck. WLAN-based pedestrian tracking using particle filters and low-cost MEMS sensors. In *Positioning, Navigation and Communication, 2007. WPNC'07. 4th Workshop on*, pages 1–7. IEEE, 2007.

[48] H. Wang, H. Lenz, A. Szabo, U.D. Hanebeck, and J. Bamberger. Fusion of barometric sensors, WLAN signals and building information for 3-D indoor/campus localization. In *proceedings of International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2006), S*, pages 426–432. Citeseer.

[49] W. Wang and IFA Yildiz. On the estimation of user mobility pattern for location tracking in wireless networks. In *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, volume 1, pages 610–614. IEEE, 2003.

[50] G. Welch and G. Bishop. An introduction to the Kalman filter. *University of North Carolina at Chapel Hill, Chapel Hill, NC*, 1995.

[51] G. Yavas, D. Katsaros,
"O. Ulusoy, and Y. Manolopoulos. A data mining approach for location prediction in mobile environments. *Data & Knowledge Engineering*, 54(2):121–146, 2005.

[52] Qing Ye. A robust method for counting people in complex indoor spaces. In *Education Technology and Computer (ICETC), 2010 2nd International Conference on*, volume 2, pages V2–450 –V2–454, 2010.

[53] Sheng Zhou and J.K. Pollard. Position measurement using bluetooth. *Consumer Electronics, IEEE Transactions on*, 52(2):555 – 558, May 2006.

[54] F. Zhu, M.W. Mutka, and L.M. Ni. Service discovery in pervasive computing environments. *Pervasive Computing, IEEE*, 4(4):81 – 90, 2005.