AN INTELLIGENT HOSPITAL NAVIGATION AND GUIDANCE SYSTEM FOR VISUALLY IMPAIRED PATIENTS

A DOCTOR OF PHILOSOPHY (PhD) THESIS

in

Software Engineering

Atılım University

by

MUSTAFA KAHRAMAN

JULY 2016

AN INTELLIGENT HOSPITAL NAVIGATION AND GUIDANCE SYSTEM FOR VISUALLY IMPAIRED PATIENTS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

ATILIM UNIVERSITY

BY

MUSTAFA KAHRAMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

THE DEPARTMENT OF SOFTWARE ENGINEERING

JULY 2016

Approval of the Graduate School of Natural and Applied Sciences, Atılım University.

_____

Prof. Dr. K. İbrahim Akman

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

_____

Prof. Dr. Ali Yazıcı

Head of Department

This is to certify that we have read the thesis "An Intelligent Hospital Navigation and Guidance System for Visually Impaired Patients" submitted by "Mustafa Kahraman" and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Asst. Prof. Dr. Atila Bostan	Asst. Prof. Dr. Çiğdem Turhan
Co-Supervisor	Supervisor

Examining Committee Members

Asst. Prof. Dr. Çiğdem Turhan	_____
Prof. Dr. Ali Yazıcı	_____
Prof. Dr. Gerhard Wilhelm Weber	_____
Doç. Dr. Hakan Öktem	_____
Asst. Prof. Dr. Gökhan Şengül	_____

Date: 15.07.2016

I declare and guarantee that all data, knowledge and information in this document has been obtained, processed and presented in accordance with academic rules and ethical conduct. Based on these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

MUSTAFA KAHRAMAN

ABSTRACT

AN INTELLIGENT HOSPITAL NAVIGATION AND GUIDANCE SYSTEM FOR VISUALLY IMPAIRED PATIENTS

Kahraman, Mustafa

PhD, Software Engineering Department

Supervisor: Asst. Prof. Dr. Çiğdem Turhan

Co-Supervisor: Asst. Prof. Dr. Atila Bostan

July 2016, 153 pages

Intelligent guidance in complex environments is critical to achieve mobility for the visually impaired, and encourage them to be more actively involved in society without spending time to learn about procedures related to indoor environments. Our system, Invisible Eyes, provides intelligent navigation and guidance to the visually impaired. The system implemented is for hospitals chosen as complex environment where procedures must be applied in order to navigate. The system allows the users to input their objectives via specially designed user interface, and destination targets are determined based on their purposes by the system. Path optimization is performed by adaptation of the travelling salesman problem, and real time instantaneous instructions are provided to users. For evaluation of the system, an environment representing hospitals is established and tested by visually impaired participants and the results show that the intelligent purpose selection-destination evaluation mechanism part of the system is approved and found to be effective by all the participants.

Keywords: Intelligent guidance, indoor navigation, RFID, real time systems

ÖZ

GÖRME ENGELLİ HASTALAR İÇİN AKILLI BİR HASTANE NAVİGASYON VE REHBERLİK SİSTEMİ

Kahraman, Mustafa

Doktora, Yazılım Mühendisliği Bölümü

Tez Yöneticisi: Yrd. Doç. Dr. Çiğdem Turhan

Ortak Tez Yöneticisi: Yrd. Doç. Dr. Atila Bostan

Temmuz 2016, 153 sayfa

Görme engellilerin dolaşımının sağlanması ve iç mekanların karmaşıklığını öğrenmelerine gerek kalmadan topluma dahil olabilmeleri için karmaşık ortamlarda akıllı yönlendirme sistemleri kritiktir. Görünmez Gözler (Invisible Eyes) isimli sistemimiz, görme engelliler için akıllı rehberlik ve yönlendirme yapmaktadır. Sistem, bina içi yönlendirme için prosedürlerin ugulanmasının şart olduğu karmaşık ortam olarak hastaneleri hedeflemektedir. Kullanıcılar, kendileri için tasarımlanmış özel kullanıcı arayüzlerini kullanarak amaçlarını girebilmekte ve girdikleri amaçlar doğrultusunda da sistem hedef noktalarını hesaplamaktadır. Gezgin satıcı problemi adapte edilmiştir ve en uygun rota hesaplaması yapılarak kullanıcıya gerçek zamanlı yönlendirme komutları verilmektedir. Sistemi değerlendirmek için hastaneyi temsilen bir test ortamı hazırlanmıştır ve sonuçların da gösterdiği üzere akıllı amaç seçme ve hedef noktalarını belirleme mekanizması kullanıcılar tarafından kabul edilmiş ve etkili bulunmuştur.

Anahtar Kelimeler: Akıllı rehberlik, bina içi yönlendirme, RFID, gerçek zamanlı sistemler

To My Wife, who encouraged and supported me throughout this process…

ACKNOWLEDGMENTS

I express sincere appreciation to my supervisor Asst. Prof. Dr. Çiğdem Turhan for her guidance and insight throughout the research. Thanks also go to my co-supervisor Asst. Prof. Dr. Atila Bostan, the members of our participatory design team, Solvent Software Company, Prof. Dr. Ali Yazıcı, Prof. Dr. Gerhard Wilhelm Weber, Asst. Prof. Dr. Erhan Gökçay, Asst. Prof. Dr. Gökhan Şengül, Assoc. Prof. Dr. Hakan Öktem, and especially to our test group for valuable contributions that greatly improved the work.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

GPS	-	Global Positioning System
RSSI	-	Received Signal Strength Indicator
SCDM	-	Signal Coverage Density Method
RFID	-	Radio Frequency Identification
LED	-	Light Emitting Diodes
NN	-	Nearest Neighbour Algorithm
SRS	-	Software Requirements Specifications
C2	-	Components and Connectors
TSP	-	Traveling Salesman Problem
PCTSP	-	Precedence Constrained Traveling Salesman Problem
ATSP	-	Asymmetric Traveling Salesman Problem
STSP	-	Symmetric Traveling Salesman Problem
MTSP	-	Multi Traveling Salesman Problem
UHF	-	Ultra High Frequency
HMM	-	Hidden Markov Models
LSE	-	Least Square Estimation

Chapter 1

INTRODUCTION

In almost all of our daily activities, mobility, which means to move or be moved freely, easily, and safely in order to reach a destination, is required. Compared with people who do not have any mobility limitations, visually impaired people have a significantly lower quality of life because of their vision limitations. Indoor positioning is a challenging field for researchers because an efficient positioning system is not developed till now for the indoor environments. Indoor navigation and guidance problem is critical when we considered the visually impaired people because the accessibility of the indoor environments is totally dependent to the success of indoor navigation and guidance systems for them.

The difficulty of mobility can be seen especially in complex environments and complex processes which make indoor navigation and guidance critical for visually impaired people. Hospitals are good examples for complex environments and mobility in hospitals are complicated processes depending on predefined procedures. Therefore, after considering these type of environments it is possible to say that intelligent guidance and navigation systems are expected to improve the accessibility and quality of life of visually impaired people. The rules and regulations related to the hospital for indoor navigation should also be considered. According to legal acts, some regulations related to the accessibility of visually impaired patients becomes a requirement for hospitals for better healthcare. Therefore, design and implementation of such systems are needed requiring an interdisciplinary work including computer science, electronics and software engineering. In the proposed work, an intelligent navigation and guidance system for the visually impaired will be designed and implemented.

The chapter aims to summarize the philosophy behind the dissertation. In the first part of the chapter, the main motivation of the work is given. The following sections are to clarify the aims and objectives, describe main methodologies for research and finally state the contribution to knowledge.

1.1 Motivation

The main motivation to conduct this study is to help the visually impaired people to be more involved in society in which they can improve their quality of life by having ability to access indoor environments. In this study effort is spent to analyze, design, implement, and validate the proposed intelligent guidance and navigation supportive system for those people in becoming mobile, feeling independent, and being a self-sufficient member of the society.

1.2 Aims and Objectives

As stated before the main objective of the thesis is to develop an intelligent guidance and navigation system for the visually impaired. The system named “Invisible Eyes” aims to provide a purpose selection mechanism in the navigation system. In this way, the navigation system can be used for intelligent guidance as well.

The initial objective is to state the main trends, achievements, and challenges in the literature related to the problem. Therefore, the focus points of the study can be defined and justified by literature review.

Defining the necessary software engineering process was also established as an objective. Since the problem involves a special target group containing visually impaired people, the process of the study must be designed in the initial stages of the work.

The problem involves many sub problems which are directly related to the main problem but are not visible. The study also aims to provide solutions via algorithms and techniques in the design of the system.

The validation of the proposed system is done by a test team including visually impaired people. The tests are all performed by directly using the target audience. Assessing the real success of the proposed system was an objective and for this aim blindfolded subjects were not included.

1.3 Research Methodology

In order to reach the desired level of quality in the research and to guide other researchers working in the same field, the path of research or research methodology is

included. Although the methodologies are given in the following chapters in the thesis, a brief explanation is given.

The literature review was critical to gather information, classify similarities in the studies, recognize the challenge, and discover the focus points. Therefore, the study was initiated by reviewing materials from textbooks, journal publications, conference papers, and other materials where related knowledge of the subject topics can be found. The main technologies, techniques, and methodologies applied to the problem are reviewed during the literature review.

After the review stage of the study, the main conclusion was that, a wide range of technologies, techniques, and methodologies have been studied resulting in varying success rates. In previous research, the main focus is given to the proposed technique, yet, the required concentration on the target audience and the design process are not presented. Therefore, our initial focus is dedicated to the target audience which is visually impaired people. A team where the target audience is at the center is constructed. The analysis, design, implementation, and validation of the study have been conducted with the constructed team utilizing the type of methodology named participatory design. Since the target audience is a special group having unique requirements, we believe that the success of any system depends on how these requirements are defined. This was the reason of deciding on participatory design methodology as the study process.

In the participatory design team the target audience, specialists, software engineers, and managers as well as a software company, Solvent Software Company, are included. The software engineers, specialists, and managers from the company have been involved in regular meetings with the participatory design team. The inclusion of the company to the research was to learn the perspective and opinions of software engineers and specialist from the information technology sector.

The stages of the work correspond to the software development life cycle (SDLC). The main steps of the SDLC starts with requirements specification where the requirements of the study are analyzed in detail. In the design stage, the design of the software depending on the requirement analysis performed. Lastly, the implementation and development of the software is performed on the build stage. In the final stage, the software is tested according to the SDLC. The purpose of the study is not solely to develop a system as a project work. The study aims to provide an intelligent guidance

and navigation system for the visually impaired depending on the special requirements of the target audience and introduce an innovative purpose specification mechanism.

The algorithms and techniques proposed in the study related to the work are innovative adaptions of the algorithms and techniques in the related literature. These methods are adapted by considering the special requirements of the study. The validation of the proposed technologies and methodologies are performed similarly as proof of concept and technology. The proof methodology is generally applied in the validation of software architecture where the concept has to be proven before any concrete implementation exists. The main parts of the method are reviewed and the related parts are partially implemented. The problem is represented by cases and these cases are reviewed by the participatory design team. Apart from the review, the proposed techniques and algorithms are also implemented. In these implementations, the predefined cases are simulated and the performance of the proposed algorithm or technique is investigated from the simulation results. The effectiveness has been the main consideration in these works.

After all the studies are completed, the system has been tested and validated. For this purpose, a test team of visually impaired people is formed. The test cases covering the main targeted purposes of the system are performed with the help of the subjects in the test team. An example indoor environment is constructed as a representative environment in which all the tests are performed. The tests are discussed in detail to determine the effectiveness of the system. The internal and external system features as well as the user acceptance of the system is questioned. Furthermore, the success of the internal system features including the proposed algorithm which are not directly recognizable by users, are investigated in some artificial cases.

1.4 Contribution to Literature

The main objective that has been achieved in this work, is the design and implementation of an intelligent indoor guidance and navigation system for visually impaired people. The system is intelligent in terms of the interaction mechanism to the users. The users are able to input their purposes of being at indoor environment without struggling to learn about the procedures and rules about the processes related to that place to reach that purpose. The required procedures are embedded in the system and the system is capable of computing the destination points given the purpose. The

mechanism for achieving this idea is designed, implemented, and validated in this work. To the best of our knowledge, there has been no comprehensive study on the analysis, design, implementation, and validation of an intelligent guidance and navigation system which focused on the needs of the visually impaired people and provided a purpose specification mechanism.

The participatory design and agile software development methodologies are followed in all stages of the work. The analysis, design, implementation, and validation stage are all performed with the help of the participatory design team.

The design and implementation of the algorithms related to the problem can be considered another contribution. The lateration algorithm is adapted for the imperfect cases where all the estimated distance circles are not intersecting. The proof of concept and technology principle, which depends on the implementation and review of the results of artificial cases, is followed in the validation of the design of algorithms and also the software architecture. For this reason, artificial cases representing the imperfect cases are simulated and results are discussed.

The validation part of the work is performed by the test team containing blind individuals. A comprehensive test procedure, including predefined cases, is defined. The test team is asked to complete the tasks in these cases by using the implemented system in a representative environment. This testing and validation is also a contribution because of using predefined cases, using a representative environment as a complex indoor environment like hospitals, and including subjects of different age, mobile device utilization skills, and white cane using skills.

1.5 Thesis Layout

The layout of the thesis corresponds to the structure of the work and follows the stages of the software development process as mentioned previously.

Chapter 2 provides the summary of the literature review and related work. The chapter includes knowledge related to the thesis work to fully understand the main aspects where general techniques, technologies, and methodologies related to the field are covered and classified.

Chapter 3 details the analysis of the study by aiming to start the work from the requirements specifications, compatible with the software development life cycle. The justification of the proposed system is given and a detailed design including the

architecture, main system components, technologies, techniques, and algorithms are presented with the involvement of a participatory design team.

Chapter 4 provides the implementation details of the recommended system including details about software patterns, the automaton view of the system, knowledge engine, and details about algorithms.

Chapter 5 gives the test and validation results along with the test purposes and methodologies.

Chapter 6 concludes the thesis with suggestions for future work.

Chapter 2

BACKGROUND

In this chapter, a literature review related to the techniques and algorithms used for indoor navigation and guidance for visually impaired people is given. Firstly, the technologies related to indoor navigation are provided, followed by a classification of generally studied technologies related to the problem. Then, the most applied and accepted techniques and methodologies are listed. In the next section, navigation and guidance studies which can be seen as a system are given. Finally, the results obtained from the literature are discussed which provide the main motivation for the study.

2.1 Indoor Positioning *Techniques & Technologies*

Widely studied indoor positioning techniques and technologies included. Image and vision based, Bluetooth related, RF based, and sound related techniques and technologies. Hybrid techniques and technologies composed of the combination of multiple techniques and technologies, can also be found in the work of some researchers. Since the techniques cannot be separated from the technologies, both are considered together.

2.1.1 Image and Vision Based Techniques & Technologies

The use of cameras for indoor positioning applications gained an enormous importance nowadays even if the idea is an old one. This improvement is because of the development in optical methods, data transmission rates and image processing techniques Mautz et al. (2011). Generally, camera based systems measure image coordinates depending on angles with the, Angle of Arrival Technique. Alternative to multiple views of a single camera, several images obtained from several cameras are preferred. In order to determine the object coordinates, a rotation matrix, a constant related to camera specifications, and a scale factor are used.

In order to obtain an estimation for depth, synthetic stereo vision is utilized, where the scene is observed by cameras in different locations. Scales estimation in this technique is complicated requiring additional techniques and lasers can be used for this purpose.

A wide range of algorithms are employed for camera based indoor positioning systems as mentioned in the work of Trucco et al. (2006). A well-known approach in the use of cameras is to try to match the objects extracted from image data to the previously modeled building objects. This technique, approximates the position of the person. For example, if we are able to identify a unique object which is near the subject whose position is in question, we can find the approximate position of the subjects since we already know the unique position of the unique object. A hierarchical approach where firstly the room of the person is identified and then the exact position in the room is estimated as presented in the work of Kohoutek et al. (2010). The exact position of the person is found by querying the database for the object and utilizing classification techniques.

Instead of a single object, a sequence of objects can be considered. Sequences of images of all possible paths are previously taken and stored. The taken image is compared with the images in these sequences as another possible technique for indoor positioning which is introduced by Muffert et al. (2010). This technique requires an omnidirectional camera system which is able to take 360° views of the surrounding area at each camera position. Researchers used an omnidirectional camera system consisting of five cameras positioned in a horizontal ring and one camera on the top. The use of the system is in robotic applications and in autonomous navigation for ego-motion which means estimation of moving position of an object relative to its surrounding environment. For example, a car's moving position corresponding to lines on the road being observed by the car is in the field of ego-motion. In order to obtain better results, authors suggested to apply their system by the conjunction with global positioning system for real-time applications.

These methods are not seen as reliable methods and researchers are continuing their studies on different, more dependable techniques. The main motivation is to improve the robustness and increase accuracy. A marker detection system, which depends on distributing coded targets to the environment and estimating the position depending on these targets, is suggested by some researchers. These coded targets can be placed on the ceiling, walls and objects. When the target is detected, the position

information is obtained from the database of targets by invoking queries to the database.

This approach is implemented by Mulloni et al. (2009) by placing coded targets on the ceiling and wall. Proposed marker-tracking software library was able to work with a camera phone's position and orientation with respect to previously established markers. In order to exploit a marker, the user must point the camera of his phone at the marker. The position of the user can be inferred with the help of these markers. Even though the proposed system is for applications like conference guide, it can be applied to the navigation of visually impaired people. However, the technical specifications of a phone camera may not be sufficient and special devices should be applied. In addition, specification of markers should also be adopted for the problem of navigation of blinds.

Visio based systems, which provide virtual reality, are also studied by many researchers. In the work of Wong et al. (2003), the virtual reality technology was proposed for blind navigation which helps the blind users to discover the environment using the technology. The proposed system contains two cameras for obstacle recognition and depth evaluation different than their previous studies which use only one camera. The system also uses a fuzzy segmentation procedure for image pre-processing. The segmented images undergo a rule-based stereo matching procedure to evaluate the distance between the user and object. The location and size information related to the object is evaluated and conveyed to visually impaired individuals by means of structured sounds. The system is suitable for static environments and does not give direct navigations.

Hub et al. (2004) suggested another camera based system that assists visually impaired people in orienting themselves in indoor environments. In the study, a system aims to determine the position of objects and individuals in an indoor environment. This solution involved the use of cameras with sensors to detect objects and the direction in which the user is moving. The visually impaired person is given a mobile device like a flashlight. On the request of the user, a connected portable computer receives the gathered data sent from the mobile device and makes calculations to evaluate the information related to the environment where the user is located. The answers coming from the connected portable computer, are via the text to speech engine through an earphone. As stated in the study, there are many challenges about the concept and these challenges must be solved before a global deployment. The

system is not easy to implement, since it requires a specialized device to enable the user to interact with the environment and the success of the system strongly depends on the success of the specialized device.

Another vision-based navigation system to guide the visually impaired users in both indoor and outdoor environments is suggested by Treuillet et al. (2010). In the study, the limitation of the global positioning system (GPS) dependent systems is emphasized and a system working both indoor and outdoor is aimed. The positioning system uses a body seated camera that continuously takes pictures from the environment. A real time working mechanism tries to extract particular references from the images coming from cameras and matches these images with 3D landmarks already stored in the system. System obtained good results to locate people in memorized paths but it was not suitable to be used in environments that are visited for the first time.

System using visual landmarks is studied by Serrao et al. (2012). The proposed system uses visual landmarks for localizing the user in the building. After the position of the user is determined, the system works for tracking and validating a route for navigation. The proposed system helps the user by improving the user's mobility and autonomy. The landmarks which are critical for the system includes staircases, fire extinguishers, doors, etc. The system integrates the data of a GIS of a building with visual landmarks. Image processing techniques are implemented for detection of stairs and doors. A prototype is developed by researchers for validation purpose and is tested at the Institute of Engineering of the University of the Algarve. Successful results were obtained but lack of visual landmarks at certain locations was a problem. Additionally, the proposed system is not tested with the target group of visually impaired subjects.

In a similar work, Moreno et al. (2012) worked on a system which aims to detect obstacles, lateral doors in coridors and classify door types. The motivation in the work was to provide blinds with cheap and easy to use tools. The proposed system only requires a smart phone with a built in camera. The system was aimed to serve for a specific purpose which is detecting and classifying the doors in a corridor. Consequently, this system could not solve the huge problem of navigating the blinds but can only be used inside as part of another comprehensive system.

2.1.2 Sound Based Techniques & Technologies

Sound which is a mechanical wave transmitted over a medium can also be used for indoor positioning. Most of the work in this field are concentrated on ultrasound. Time of arrival measurements are used to obtain the distance between emitter and receiver. The time of arrival method uses the absolute times at which the sound pulses arrive in a certain station.

Active and passive systems are studied for indoor localization. In active systems, the transmitter is located on the user and receivers are fixed on the building. Position of the user is evaluated by the receiver's network. In multiuser systems, signals cannot be identified uniquely and some related problems arise. On the other hand, in passive systems users are attached a receiver and the multiuser cases are not considered problematic anymore.

One of the first active systems was the one published by Ward et al (1997). Users are attached with a transmitter and receivers are placed at known positions around it. The distance between transmitter and receivers are calculated by time of flight of sound pulses from the ultrasonic transmitter to the surrounding receivers. The transmitter's location can be found after all the distance values are calculated with the lateration algorithm. The working principle of the proposed system has similarities with bats. The system depends on the journey of sound pulses from transmitter to the receiver and the physical conditions of the surrounding environment makes a dramatic impact on the pulses which result in errors in the position calculations.

In a more recent work, a range measurement technique is published by Sato et al. (2011) with a high accuracy rate. A motion capture system using ultrasonic communication is proposed and compared with the existing motion capture systems. An algorithm named Extended Phase Accordance Method is devised for estimating the distance of a moving object. In the work, the use of Kalman filter is also suggested to improve the position accuracy by compensating for the angles coming from different angles.

Echolocation can also be used for indoor positioning without requiring any tags as stated in Wan et al. (2010) experiment in which a high accuracy is obtained. The hardware used in the study was a sonar module manufactured by Devantech Inc. having two transducers for transmitting the ultrasound and listening for the echo coming back. The system is tested using six ultrasonic modules in a perfect

environment. Although the results were promising, the success of the proposed system in imperfect environments must be investigated.

2.1.3 Wireless Network Based Techniques & Technologies

Wireless networks can be used to estimate the location of a mobile device within a network where signal propagation is the key concept for location estimation. Propagation of signals can be modeled or estimated depending on previous measurements.

The fingerprinting method will be discussed in the next section but as an example of wireless network positioning, the indoor path model studied by Chrysikos et al. (2009) can be considered as an accepted model using fingerprinting. Empirical fingerprinting depends on previously measured data necessary for calibration. Signal strengths are measured in different locations in the building and stored in a database. Success of the technique depends on the quantity of the calibration data and site specific estimations should be worked as stated by researchers for future work.

Active user participation can also be involved as suggested by Park et al. (2010). An organically constructed localization system depending on the idea of relying on user inputs for constructing a database, is implemented and challenges of the implementation are discussed. Limitation and challenge of the method is because of the change of environment requiring recalibration. Another drawback is that network protocols and hardware are not designed for indoor localization and RSSI values change among vendors.

2.1.4 Bluetooth Based Techniques & Technologies

Bluetooth indoor positioning is used for tracking people to provide location dependent information. The general architecture used in these studies are composed of a fixed architecture of Bluetooth beacons which include inexpensive materials and are easy to deploy.

In the work of Feldman et al. (2003), the experimental evaluation of a Bluetooth positioning system is presented by implementing the system in a Bluetooth capable handheld device. The associated distance between sender and receiver is measure by Received Signal Strength Indicator (RSSI) measure. The triangulation

method is merged with least squares estimation to determine the location. The location can be estimated with a precision of 2.08 m. The authors suggested to use the system with the combination of another location estimation method and to develop an adequate Kalmann filter to improve the precision.

In a similar work, Chawathe (2008) described a method for determining the position of a mobile device. The use of inexpensive commodity devices for beacons was the important design criteria for authors. A Bluetooth adapter powered by a USB hub is used as a prototype Bluetooth beacon. The device discovery process in Bluetooth localization is found to be a slow process. Therefore, the process should be terminated early, after a few beacons have been discovered, as stated in the study which may result in a loss of localization accuracy.

A similar working principle can be reversed to a client server architecture in which beacons scan for moving mobile devices and report to a central server. This architecture does not require the deployment of any specialized system in mobile devices. The Bluepass system suggested by Diaz et al. (2010) is an example of this architecture. The signal coverage density method (SCDM), which does not need to calculate the user coordinates and tries to determine the room where the user is probably located, was found to be suitable for the proposed architecture. The impact of changing fixed station positions, the performance of the SCDM with combining the trilateration method, and evaluation of the study by deploying and testing in different indoor environments are listed as interesting directions for future research.

In the work of Anastasi et al. (2003), the central server architecture is also proposed. A system named BIPS, which covers the building and offers services to allow a mobile user to recognize the shortest paths from his/her location to the other user's location inside the same building, is mentioned. The authors suggested to dedicate two separate hardware: one for device discovery and the other for exchange of data among BIPS and the other connected users.

2.1.5 RFID Based Techniques & Technologies

Radio frequency identification (RFID) system consists of a reader with antenna and tags which can be active or passive. Data containing unique serial numbers, is transformed from tags to the receivers. Active RFID contains scanners placed in buildings and active radio transmitters. Location estimation is mainly carried out by

fingerprinting on RSSI. Passive RFID depends on inductive coupling which means they are activating when triggered by a reader. The detection rate is limited but the required tags are small, inexpensive, and does not need a power supply.

Indoor positioning techniques and research are also applied for the case of visually impaired people as stated below. The techniques and devices mainly depend on the active or passive indoor positioning techniques.

Passive and active RFID technologies are investigated by researchers in this field such as Alghamdi et al. (2014). Received Signal Strength Indicator (RSSI) technique to position the user is investigated and an active RFID system is introduced. A mobile reader is attached to the user and active tags are distributed to ceilings of the building. Instead of directly positioning after RSSI measurements, threshold levels are described and used. The proposed algorithm scans by using different threshold levels.

For the positioning of the reader, at least three tags are needed. The system is tested considering some predefined scenarios. False negative and true positive values are calculated. These values represent the success of the system. Each correctly perceived tag means a true positive. Similarly, each false perceived tag means a false negative. Proposed power attenuation technique is evaluated to be reasonable for indoor and outdoor positioning.

In the works of Öktem et al. (2008, 2010), the researchers suggested a dynamic RFID system for a room which has fixed obstacles and setup. Three RF receivers are fixed around the room which is divided into grids. These grids are previously defined and the signal strength in each grid is measured and saved. The Bayesian Decision Theory is used for position estimation depending on the previously calculated probabilities.

In the work of Tsirmpas et al. (2015) passive RFID technique is studied. Corridors and all the reachable area is divided into grids. RFID tags are assigned to the floors addressing each grid. Client - server architecture is used. The user is equipped with the wearable component and walks through corridors by the help of the navigation instructions coming from the server. The system detects the obstacles, which can be classified as expected and unexpected, and informs the user. Dijkstra algorithm is used to obtain shortest paths between two points. RFID readers are attached on legs of the user. An ultra-sonic range finder is used to detect the obstacles on the path. To evaluate the proposed system, four scenarios are developed considering the obstacle types and four blindfolded volunteers participate in different scenarios.

The system is tested in a predefined test area. The success of the volunteers on each scenario is evaluated.

A similar navigation system which is designed on a white cane for the visually impaired is studied by Seto and Magatani (2009) by including colored navigation lines as well as RFID tags which results in a successful hybrid system.

The system proposed by Na (2006), named Blind Interactive Guide System, consists of a mobile device, a RFID reader, and required number of tags deployed in the floor. Using these elements, the system can recognize the current location of the user holding the mobile device. Then the system calculates the direction of the user. Following that, the system recognize voice commands given by the user and deliver appropriate voice messages. The need of reconfiguration of the positioning area, and the limitation to a predefined paths are mentioned as the weakness of the proposed system.

A passive RFID technology is investigated and evaluated by Nikitin and Rao (2006). Their result highlights the importance of device and propagating environment in the success of system.

A hybrid approach using passive and active RFID is also investigated by Koch et al. (2007). The proposed system is suitable for small living areas.

GPRS network addition to RFID system is used for server connection in the work of Chumkamon et al. (2008). The system tracks the person and gives new commands when he or she is lost. In this system the walking stick was not suitable for blinds, and communication delays were also a problem.

2.1.6 Promising Techniques & Technologies

Some off-the-shelf products and technologies are also employed for supporting blind navigation. Filipe et al. (2012) suggested to use Mikrosoft Kinect for this purpose. Device is specially designed to recognize user motion by involving integrated depth and color sensors. Depth images are constructed by this device and are also used for object recognition purpose. Neural networks are used to train the system for object recognition. Similarly, mobile devices can be used to measure RSSI and by this way, localize people as stated by Au et al. (2013). With proposed model, a novel indoor tracking and navigation system can be implemented on a mobile device with limited resources.

A new design as an electronic travel aid is proposed by Prattico et al. (2013). The system is composed of infrared sensors, a microcontroller and four vibrating motors used to give signals to the user when an obstacle is intercepted. All these components are integrated in a belt. User can differentiate the front and back obstacles by vibrating motors which are located two on the front and other two on the back. Sensor characteristics and physical design are studied. A low pass filter is adopted. System is tested for validation purposes. Different trials considering the obstacle type and position are performed during the test of the system and successful results are obtained.

Park et al. (2014) suggested a localization system using infrared Light Emitting Diodes (LEDs) for visually impaired people. Beacons are fixed on reference points. Users have mobile receivers and distance and orientation can be perceived by these receivers. In their study, differential intensity method using the infrared intensity level was suggested. A similar approach was previously suggested by Sonnenblick (1998). Author implemented a navigation system for indoor environments based on the use of infrared LEDs. Such LEDs must be strategically located in places used by the blind person to perform their activities.

Infrared technology is merged with Bluetooth technology by Guarrero et al. (2012). The system uses few components and accessible technology but was appropriate for small environments.

In another study, authors aimed to develop a way of finding application for visually impaired in outdoor environments starting from the design of the user interface in the work of Rondriguez-Sanches et al. (2014). After testing the existing user interfaces, they concluded that these software are not enough for visually impaired users and the authors developed a user interface for blinds. After they designed the suitable interface, they designed the path finding application. The aim of this application is to guide visually impaired people from one place to other providing required navigation commands.

A voice oriented guidance system is proposed by Moulton et al. (2009). Although object oriented architecture is well developed and a nice architecture is given the system depending on GPS is not able to give a detailed navigation.

In the work of Elbes et al. (2013), authors present a top-down design approach for a social collaboration service designed for the blinds. They also present a precise localization approach for indoor and outdoor environments. A software architecture

for social collaboration network is proposed in their work. This network provides assistance for the blind and visually impaired while they travel from a source to destination. This system provides precise position and orientation information and continuously sends it to the collaboration server. Server provides textual or audio assistance to the users. Users can interact with nearby users and share experiences or information. The proposed architecture contains a rule-based expert system. This rule based system analyzes all the rules that are related to the user's current location and orientation and provides the proper assistance to the user by giving navigation related suggestions.

Because of the complexity and interdisciplinary nature of the problem, some research groups were established as in the work of Kammoun et al. (2012). The aim of working with a user group was to construct a relation between researcher and end user. In order to extract related information from the environment, stereovision methods were used. The results obtained from analysis of the environment, is merged with data obtained from global positioning system for outdoor localization. Project was aimed for outdoor environment but suggesting to apply in indoors also. The system is tested for university campus by some previously defined scenarios and the results were significantly successful.

2.2 Common Models and Methodologies for Indoor Positioning

In this section, common models applied for indoor location estimation is listed. Received signal strength indicator which has a huge importance for location estimation systems is mentioned first, followed by trilateration, fingerprinting, neural networks, and soft computing are mentioned next.

2.2.1 Received Signal Strength Indicator (RSSI)

Received Signal Strength Indicator (RSSI) is the measurement of the received signal strength power. The indicator is used for distance measurement for proximity applications and coarse-grained location estimations, by applying theoretical models. A wide range of products can be purchased off the shelf could be used for RSSI measurements without requiring any additional hardware. Therefore, the implementation of received signal strength is simple as stated in Rozyyev et al. (2010).

2.2.2 Trilateration Technique

Although many variations are studied, the trilateration algorithm is the base algorithm which can be defined as the process of determining the relative position of objects by using the geometry of circles obtained after distance measurements. In the multilateration technique which is initially developed for military applications, the position of the aircraft is determined by measuring time of arrival of signals from two ground stations. The trilateration algorithm is used in some of the studies for position determination but when the technique is applied for Bluetooth beacons, the error in signal propagation appears and because of the signal strength estimation algorithms, the technique may not be applicable. Therefore, modifications to the algorithm is needed in these cases.

2.2.3 Location Fingerprinting

Location fingerprinting is modeling signal propagation by constructing a radio map of previously recorded measurements from different indoor locations. The strength of radio signals are generally used as type of measurements in fingerprinting. The location map containing fingerprints or map of measurements are recorded for the building and when a new measure denoting the position of the user is asked, the system estimates the position by employing the previous measurements.

The problem of deriving the position from the signal strength can be considered as an inverse problem and a tool adopted from Statistical Learning Theory can be used to model the functional dependency between the position and signal strength as stated by Brunato et al. (2005). The proposed Support Vector Machine algorithm showed a very low error rate compared with other three algorithms which are weighted k nearest neighbors, Bayesian modeling, and multi-layer perceptrons. The results of the Support Vector Machine algorithm was very close to the k nearest neighbors algorithm. Kalman filter based training of a neural network, designed for fingerprinting, is studied by Takenga et al. (2004) and even if the observed performance is not better than the Gradient descent training of neural network, the training effort is decreased which is significant for cases where large neural networks are required.

Variety of different models are studied but the difficulty is in the representation power of these models. RSSI is strongly influenced by the environment and modelling in reality is not so successful with this approach.

2.2.4 Neural Networks

Pure use of trilateration technique for indoor location estimation is difficult for obtaining the accurate location and so efficient classification and clustering artificial techniques like neural networks (NN) which is capable of tracking noisy measurements are applied to the problem by many researchers.

Multiple neural networks are applied with Bluetooth indoor localization in the work of Altini et al. (2010). The proposed low-cost Bluetooth based localization system, achieved high precision of accuracy by taking into account the user's orientation.

2.2.5 Soft Computing

Different than neural networks, the soft computing methodologies are also applied to the indoor location estimation problem. In the study of Yun et al. (2009) a system employing fuzzy logic with genetic algorithms is proposed. They observed improved performance compared to the existing methods. Fuzzy logic technique is also studied for child tracking applications in the work of Rozyyev et al. (2011).

2.3 Results of Literature Review

Literature review is considered as the first step of the work in order to gather knowledge of the field by identifying the missing or untouched parts. Reviewed studies are classified in term of technology and techniques applied or proposed. These techniques and technologies can be classified as image and vision, sound, wireless network, Bluetooth, and RFID based. Apart from these, some other studies including hybrid techniques or describing complete system are included. The last part of the chapter was devoted to the main methodologies including lateration, RSSI, location fingerprinting, neural networks, and soft computing to present a picture of the general strategies applied to the indoor positioning problem which is in the focus of guidance and navigation of blinds.

Hybrid applications which are combination of techniques improved the performance as stated in the studies. This result motivated us to consider different

technologies in our work. Additionally, a need of a sufficient, innovative, and robust technique for indoor positioning is obvious although not aimed in this study.

The main result of the literature review is the two focus points of our work: the need of a well-planned working methodology which best suits the problem and the design of an intelligent guidance and navigation system. In most of the studies, all the concentration was given to the proposed technique, yet the planning of how to conduct the study, which is very critical when we consider the target audience, was omitted. Therefore, in the next stage of the work, we considered this finding and worked for a methodology which is based on working together with the target audience and involving them in all stages. The second finding of the review was the need of an intelligent guidance and navigation system which has not been considered in previous research and has become the main motivation of our work. The work is aimed to design, implement, and validate an intelligent guidance and navigation system with suitable working methodology which puts the target audience in the center.

Chapter 3

ANALYSIS AND DESIGN

This chapter explains the analysis and design steps of the "Invisible Eyes" system in detail. The most crucial part of the analysis and also of the software development process is the requirements analysis stage. Requirements analysis includes the tasks of determining the needs or conditions to meet the desired product, taking into account the needs of all the participants of the project, as well as analyzing, documenting, and managing the software requirements (Kotonya, 1998). The first part of the chapter is devoted to the user requirements and constraints. In this phase of software development, all the participants including users, developers, specialists, and managers are included according to the participatory design methodology which is explained later in the chapter as a software design methodology for special projects like Invisible Eyes. Rules and regulations related to the study are also considered and some of them are included as constraints. In addition, for all the basic requirements, use cases are given.

After listing requirements specifications, the software design stage starts and the participatory design team is included in this stage as they were in the analysis, implementation and test stages. In the software design stage, specifications of the software artifacts for defined software requirements and constraints are constructed for an intended environment (Ralph, 2009). The system is divided into two sub-systems, namely, the main system and the user interface system, and these main components are designed. Software architecture is also explained in this chapter and architectural styles and patterns are detailed. Similarly, software design patterns for the system are also mentioned.

Apart from requirements, constraints, and architectural considerations, the design of the main algorithms of the system is also mentioned in this chapter. As stated in the previous chapters, the focus of the study is to propose an intelligent guidance and navigation system. The user utilizes the system by inputting his/her purpose of being in the building and the system converts the given purpose to the required destination points providing intelligent guidance. The algorithms and mechanisms of

this intelligent component of the system are described in detail at the end of the chapter.

3.1 User Requirements and Constraints

Defining user requirements for indoor positioning and guidance system is a critical element for the design of the system as well as to justify the research and development in this field. User requirements are not considered in detail in the literature and this is why most of the research is only theoretical and not implemented in real life conditions. Therefore, defining user requirements and constructing the research around these requirements will provide valuable contribution to the literature.

Even though the purpose of this work is only to develop the prototype of product, still requirements are crucial because the proposed intelligent guidance and navigation system is aimed to increase the life standards of visually impaired people if utilized in other products. Requirements engineering gains an increasing importance because of the changing nature of industry and society as stated by Berenbach et al. (2009).

Solvent Software Company is also interested and included in this work. The company mainly develops hospital information systems but also deals with products and services related to impaired people. The proposed intelligent guidance system was approved by the company and they decided to support our work. Apart from the company, target users were also included in the study. One of the visually impaired institution named "Aktif Görme Engelliler Derneği" was invited to contribute in different phases of the project. One of the members of this institution, Recep Kısacık, was involved in all of our weekly meetings. He always come up with new ideas and was always excited about our progress.

As stated before, participatory design was utilized as our methodology where software engineers, technical specialists, project managers, and Recep Kısacık are involved.

Requirements are obtained as the first step of the participatory design work. End users, shareholders, technical specialists and software engineers worked together to clarify the requirements. In the center of the work, user requirements are placed.

3.1.1 Analysis with Participatory Design Team

At the beginning of the project, regular weekly meetings with the team were scheduled. During the initial meetings, members shared their knowledge related to the field. In the later meetings, members contributed by giving their opinions about the generated ideas and scenarios, where the user requirements were the first consideration.

Apart from the user requirements, regulations and rules related to the accessibility in indoor environments were also considered when defining requirements. User requirements must consider the rules and regulations described in T.C. Aile ve Sosyal Politikalar Bakanlığı (2015) and T.C. Sağlık Bakanlığı (2015). Based on the feedback received from regular team meetings, rules and regulations related to accessibility in buildings, and rules and regulations related to public hospitals the following basic requirements can be listed:

- Required physical properties of indoor environment is defined in detail. For all the horizontally accessible point in the building at least 110 cm wide paths without any obstacles must be provided for the visually impaired people.
- For public hospitals, at least one policlinic room must be devoted to the visually impaired people, at least two (one for male and one for female) rooms must be devoted to visually impaired for each department, at least two (one for male and one for female) restrooms must be devoted to visually impaired in policlinics, and entrance of the hospital buildings and all the paths must be designed according to the accessibility rules.
- According to the weekly meetings, the mobile devices involved in the system must have accessibility options and user interface for the visually impaired.
- Users must be able to navigate in a path, and choose a purpose.
- Apart from the navigation, the system must also have guidance capabilities such that the surrounding environment must be informed while walking in the corridor.
- Also, when the user is in the wrong direction and the user goes out of the accessibility path, the user must be informed.
- In an emergency, the user must be able to communicate and get guidance from a person located in the building.
- The system must give intelligent guidance by considering procedures, dynamic changes related to navigation, and user related information.

- Before using the system, the user must be trained in a training track with the guidance of a specialist in the building.
- Some items related to the differences in physical and technological conditions of the buildings are also discussed. The system must not be strongly dependent on the network infrastructure of the building. The system should work without any dependence to any of the building infrastructure for a sufficient period of time.
- Additionally, the system must inform the user when the system usability, reliability and safety decreases.

Use cases, defined below were generated based on these requirements dealing with route generation, destination specification, navigation, etc.

3.1.2 Requirement and Constraint Specifications as Use Cases

Software Requirements Specifications (SRS) establishes the main items of an agreement between contributors on the features of the product as stated by SWEBOK Version 3, Bourque (2014). The main requirements are expressed as use cases. Considering the listed requirements generated by the participatory design team, requirement specifications are given in the rest of the section as use cases.

Use case: ***Specify Purpose***

The user must be able to input his/her purpose for being inside the building. The purpose may include a condition. Before this use case can be initiated, the user should hold the mobile device on his/her hand and system starts. The steps followed by the system are given below:

1. The user selects his/her purpose by using provided user interface.
2. If the purpose has conditions, the user similarly selects the appropriate one.
3. The system informs user about the progress and gives next instruction.

Use case: ***Evaluate Route***

The system should be able to evaluate the route given the purpose. The route is evaluated by using predefined rules. The system should be able to utilize the rules to determine the destination points. Additionally, the system should be able to put the destination points in order such that total travel distance is minimized. To evaluate the route the following steps are followed:

1. The user ask the system to evaluate the rout after specifying the purpose.
2. The system determines the destination point by using predefined rules.
3. The system orders the destination points in an optimum manner such that the total travel distance is minimized for the user.
4. The system informs the user about the progress.

Use case: ***Inform about Destination***

The system should provide needed information about each destination point to the user before starting to navigate the user to this destination point about what the next destination point is and why this destination is in the route. The steps involved are:

1. After the route is evaluated or previous destination point is reached, the system provides the required information to the user related to the next destination point.
2. The user can approve or skip the next destination point.
3. If the user approves the next destination point, the system starts to navigate user to that point. Otherwise, the system continues with the next destination point, if exists. If the destination point was the last one on the route, the system informs user about that the route is completed.

Use case: ***Skip Destination***

The system should provide the option to the user to skip the current destination point. The steps followed are:

1. The user can choose the option to skip the current destination point before starting navigation to that point or in any stage of navigation.
2. When the user decides to skip the current destination point, the system continues with the next destination point, if exists, otherwise, informs the user that the route is completed.

Use case: ***Re-evaluate Route***

The system should provide the ability of re-evaluating the existing route when the user leaves the route. The user must still be in the limits of the accessibility path. The required steps are:

1. The system traces the position of the user and recognize when the user leaves the current route.

2. The system informs the user that he/she is out of the route and the system evaluates the new route for the same destination point.
3. The system re-evaluates the route for the same destination point starting from the current position of the user.
4. The system starts to navigate user to the destination point considering the new route.

Use case: ***Pause/Continue Navigation***

The system should provide option to the user to pause the navigation on to continue the paused navigation. The user must be able to choose the option in any phase of the navigation. The required steps are:

1. In any stage of the navigation, the user can choose the pause option.
2. The system freezes the navigation.
3. The user can choose to continue the paused navigation.
4. If the user chooses to continue a paused navigation, the system continues to navigate the user considering the current position of the user.

Use case: ***Cancel Route***

The system should provide the option to cancel the current route. In any stage, the user can decide to cancel the current route. The steps followed are:

1. The user decides to cancel the route and selects cancel route option.
2. The system cancels the route and deletes all the computed data related to the last route.

Use case: ***Inform when on the Accessibility Path***

The system should inform user periodically when the user is inside the borders of the accessibility path. The information given in such case should be a simple positive stimuli. The required steps are:

1. The system recognizes that the user is inside the border of the accessibility path.
2. The system gives the positive stimuli to the user.
3. The process should repeat in short time intervals.
4. When the user is out of the borders of the accessibility path, the system must stop giving positive stimuli.

Use case: ***Inform when leaving the Accessibility Path***

The system should inform user periodically when the user is outside the borders of the accessibility path. The information given in such case should be a simple negative stimuli. The steps involved are:

1. The system recognizes that the user is outside the border of the path.
2. The system gives the negative stimuli to the user.
3. The process should repeat in short time intervals.
4. When the user is in the borders of the accessibility path, the system must stop giving negative stimuli.

Use case: ***Find Position of the User***

The system should be able to find the position of the user in an indoor environment. The required steps are:

1. The system continuously seeks the position of the user. If the exact position of the user cannot be determined, the approximate position is evaluated.
2. If the system is not able to determine the position of the user, the navigation stops and user is informed about the situation.

Use case: ***Navigate back to the Accessibility Path***

The system should be able to navigate the user back to the accessibility path when not on the accessibility path. The accessibility path is the safe path for the user and according to the rules and regulations related to indoor accessibility, safe roads must be provided in buildings. Therefore, when the user leaves the accessibility path which is on the safe part of the building, the system must be able to give navigation instructions to put the user back on the accessibility path. The steps followed are:

1. The system determines the position of the user.
2. The system evaluates the navigation instructions to put the user back on the accessibility path.
3. The system sequentially gives the navigation instructions and traces the progress of the user.
4. The process defined above iterates until the user reaches the accessibility path.

Some constraints were also identified by the participants mainly depending on rules and regulations. The main constraint is related to the accessibility path. Accessibility path must be constructed in the safe part of the corridors and obstacles

must be avoided on these paths. The width and maximum allowed slope of the path is also defined in rules and regulations and can be considered as a constraint.

3.2 Participatory Design

Participatory design is a methodology which actively involves all the participants including employees, partners, and users, in the design process, Norman (1998). The main purpose of the participatory design is to assure that the resulting system will meet the needs and be usable. Participatory design starts with the analysis of user requirements instead of the technology, methodology or techniques as stated by Norman (1998) where the participants are involved in all of the stages of the work. Participatory design and agile development go well together as mentioned by Kautz (2010). Consequently, in the design, development and test stages of Invisible Eyes, the combination of the participatory design and agile development are adapted as the working methodology.

Speedplay, is a participatory design method which integrates agile software development techniques as mentioned by Ferrario (2014). The key motivation behind the *Speedplay* process is to enable the development of socially important projects. Inspiration has been drawn from agile development, and *Speedplay* aims at some values and principles such as flexibility, informal collaboration, and working code. *Speedplay* was developed from the need of working in partnership with stakeholders and end users to deal with social issues within short time periods.

Speedplay is divided into the following steps. The *prepare step* outputs research methods to gather initial user requirements. The focal points are deployed at the end of this step resulting in scheduled group meetings. In parallel, our team also spent hours to define requirements of the project as the initial step. The focal point is to produce an intelligent guidance and navigation system and the *design step* aims to visualize and design systems that can address the user needs of such a system explained in the next sections. The starting point is the requirements and all the design steps depend on these requirements. The *build step* includes an agile approach with short development cycles. The *build step* refines the user requirements and concludes with the release of a stable technology prototype. The next chapter is related to the implementation of the system which corresponds to the *build* step of *Speedplay* process. As stated before, the participants are involved at every step of the process,

especially in the *sustain step*. The *sustain step* aims to establish wider partnerships to support long-term development and deployments of the prototypes. The sustain step of the *Speedplay* is not the consideration and aim of the work. However, at the end of the work, providing strong results for further work and for sustain step is aimed. These steps are visualized in Figure 1.

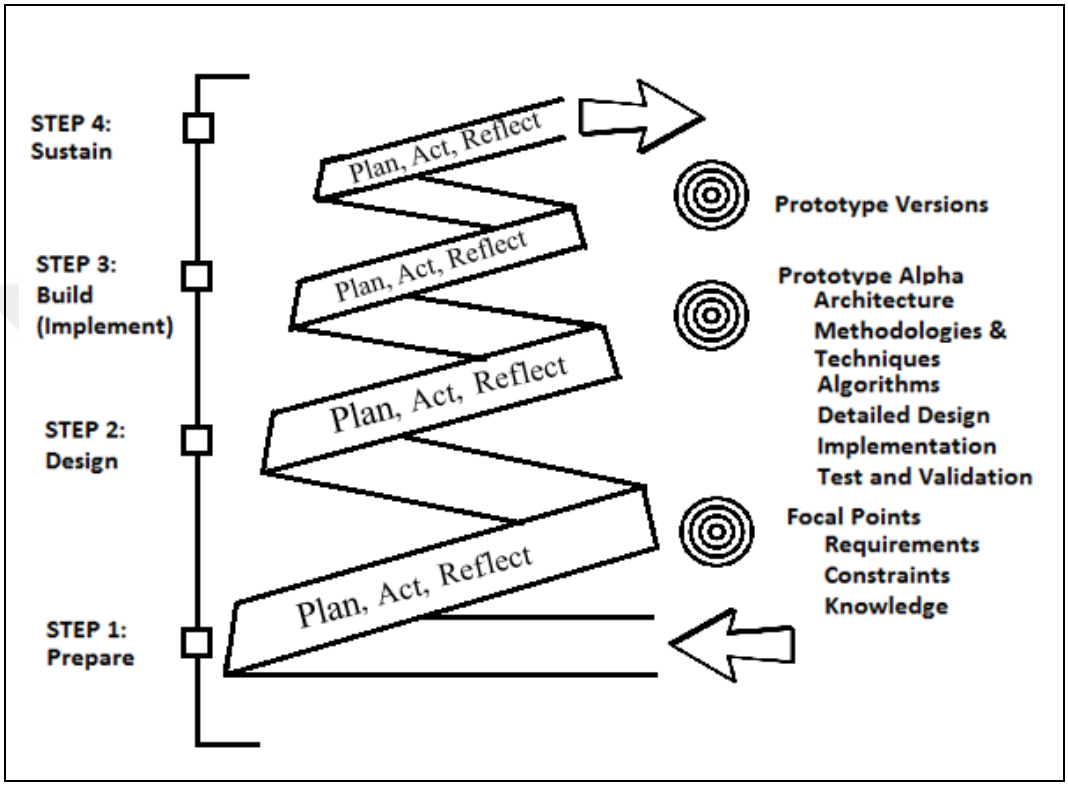


Figure 1: Representation of Speedplay participatory design technique for Invisible Eyes. Adapted from the work of Ferrario (2014).

Speedplay, was applied in the Access ASD (AASD) project. A crowded group including multidisciplinary team of academics, care service managers from a local authority, mental health therapists and practitioners from national autism charities came together for the project. At the end of the project, successful prototypes were developed as discussed by Ferrario (2014).

Participatory design approaches have been applied in health informatics discipline for a while. The project by Sjöberg (1994) aimed to develop an example of how modern computer network technology could be used to support teamwork and organizational learning in health care.

Participatory design with combination of agile development methodologies best suits our project. A design without the contribution of all participants including

visually impaired end users, technical specialists, project owner companies and software engineers will not achieve a successful result and prototype. In our system, the user requirements and constraints are determined with participatory design methodology as described in the previous section.

3.3 System Design

Considering the requirements and constraints defined in the previous sections, providing a safe accessibility path, positioning the user in indoor environment, evaluating the front direction of the user, tracing the progress of the user, and intelligent guidance are crucial for the success of navigation and guidance of the user. Therefore, these are the main considerations in the design of the system.

Since the user is not allowed to make free movements because of the safety reasons as described in the previous sections, accessibility paths on the safe parts of the building are designed. These paths are not physically perceivable and can only be perceived by the system. Designed accessibility paths are aimed to replace the well-known physically perceivable yellow lines for visually impaired people. The paths constructed with these yellow lines are not suitable for indoor environments. This is the main reason of our decision to change traditional accessibility path containing yellow lines with our digital accessibility path.

Two positioning mechanisms are designed in the system. Since the user must be navigated on a safe accessibility path, the user position on the path must be determined by the system with high precision. For this purpose, passive RFID is found to be the most appropriate technology. The system should include a reader and passive tags can be placed on the accessibility path considering the reading range. This idea was also applied in other works as described in Background chapter.

Apart from positioning the user on the accessibility path, the system must also be able to find or estimate the position of the user out of the accessibility path. Indoor positioning is a popular topic and even though there are promising studies, still a valid and accurate technology has not been developed. Majority of the studies are on Bluetooth indoor positioning. This technology is affected by the environmental conditions but approximation algorithms and techniques can be used to increase the precision. Therefore, in our work, we decided to design and develop a Bluetooth indoor positioning mechanism in addition to RFID based positioning. This hybrid positioning

mechanism will provide the system with the ability to find or approximate the position of the user when the user is on the accessibility path or out of the path.

The main contribution of the work is introducing an intelligent guidance system through the navigation. This idea is new to the literature and has not been studied before in any previously developed system based on the literature survey conducted. The intelligent part of the study was discussed in detail in the regular meetings of the participatory design team and approved by all the members. Furthermore, as another contribution of our study, the design and implementation is based on participatory design and agile development methodologies where the focus is on the requirements and contribution of the users and, design and development includes short cycles. This design and development methodology has not been applied before for indoor navigation and guidance of the visually impaired.

The system must be a real time system because the progress of the user must be traced and the system must behave according to the progress of the user. Response of the system must be guaranteed within a limited time and the system behavior completely depends on the user progress.

The direction faced by the user is also crucial for the system. A compass can be used for this purpose. The orientation of the building can be found and recorded. The heading of the user can be compared with the orientation of the building, and the looking direction of the user can be evaluated by this way.

The proposed design contains two mobile devices (in the hand and on the waist), UHF RF reader, headphone, Bluetooth beacons, and passive tags attached on the predefined accessibility path. The composition of the system is represented in Figure 2.

The design is decomposed into several modules and these modules can be deployed into two mobile devices. According to the discussions with participatory team, the separation of the user interface from the main system is needed. Since the system must also needs to evaluate the front direction of the user, the front of the mobile device must conform with the front of the user. This constraint will limit the mobility of the user and if the user does not obey the constraint of the system, the front direction will be evaluated erroneously, and consequently provide wrong navigation instructions. Therefore, by separating the system into two devices the user will not be restricted to hold the device in a restricted manner. Basic functionality of these two devices are as follows:

Mobile device on the hand of the user:

- It is a user interface and is connected with the headphone by Bluetooth connection.
- All the user interaction (input and output) is achieved by this device.

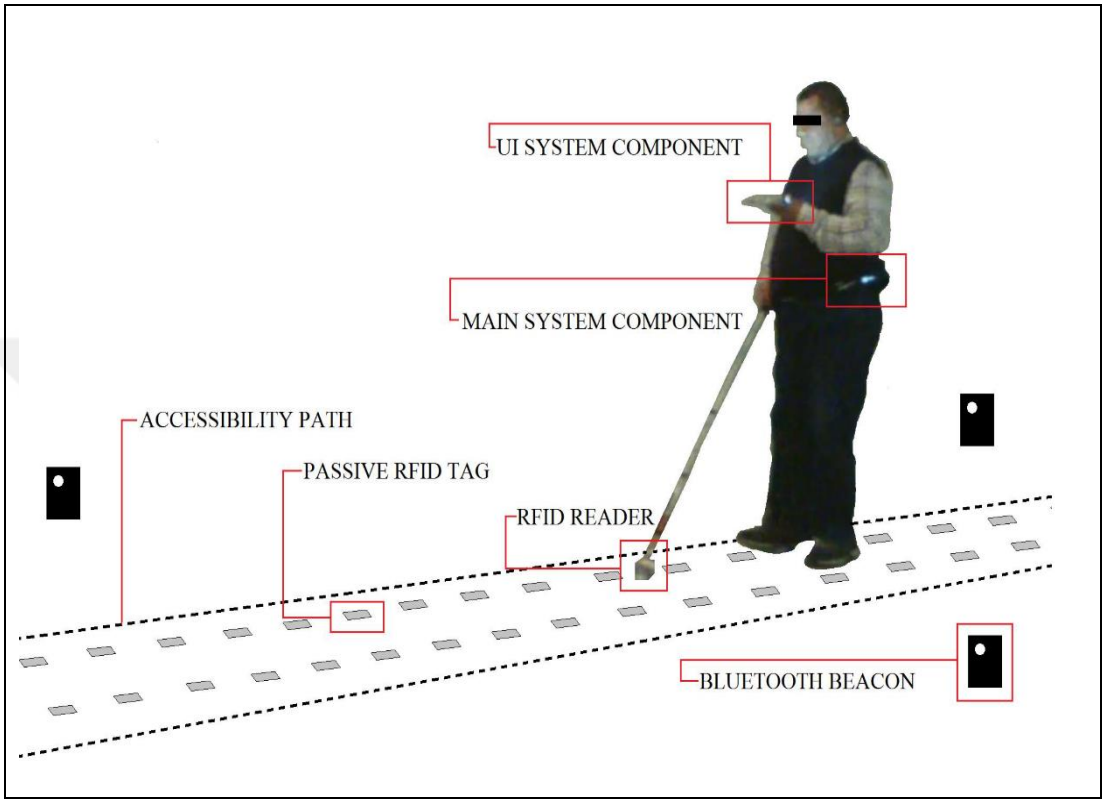


Figure 2: Visual representation of the system.

- There is a Bluetooth connection between mobile device at hand and mobile device on the waist.
- Input data and commands are sent to the other mobile device and also obtained results are received from the other device.

Mobile device on the waist of the user:

- All the sensors are connected to this device and this device is the central part of the system.
- RF reader is attached and the device receives data from the reader by a thread. Tags are identified and the position of the user on the accessibility path is determined. For position identification, classification by voting is used for the RFID system. As described in the next sections, six RFID tags are assigned to

a single graph node and user position is assigned to a graph node according to a classification algorithm.

- Compass is attached to the device. The direction of the user is continuously measured by a thread.
- A module is responsible for Bluetooth beacon. Bluetooth beacons are perceived by this module and position of the user is determined. Additionally, labels on Bluetooth beacons are perceived which are to be used for guidance purposes. For position determination, lateration method is used which is the process of determining the location of a point by measuring angles to it from known points as described in the next sections.
- Rule based knowledge engine is located on a database on the local server of the device and a module is responsible for information generation based on this server.
- Destination generation mechanism is responsible for evaluating and putting in order the destination points given the purpose. This mechanism is dependent to the rule based knowledge engine.
- Shortest path algorithm is implemented on a separate module located on the device.
- All the operations of the user are logged to a database located on local SQL server by a separate module.

The position of the user can be determined by a hybrid approach containing the data coming from RFID system and Bluetooth beacons. According to the requirements defined in the previous chapters, the navigation of the user is on the accessibility path. Therefore, the exact position of the user on the path can be measured by the RF reader and, the user must be navigated on this path by the system. However, an additional indoor positioning technique is required for safety reasons when the user is out of the accessibility path or in emergency cases as described in previous chapter, namely, the Bluetooth indoor positioning technique. Several Bluetooth beacons can be attached on the walls of the building. The configuration of the Bluetooth beacons must be carefully done considering the range of Bluetooth beacons and environmental conditions like dimensions of the corridors and shape of the rooms.

3.4 System Architecture

Because of the hierarchy and scope of components, a layered architectural style is predicted. It is necessary to classify components which has the same hierarchical level and scope on the same level in the layered architectural style. Other architectural styles are also considered and investigated but the layered style was the most appropriate for our needs. First, client-server style is not appropriate because of that the system is required to work as a standalone system. Therefore, a separate central server and client applications connected to the server does not meet the defined needs. Shared memory style and blackboard style are not chosen for the same reason. A central memory or blackboard is not possible when the constraints and requirements are considered. In addition, pipe-filter style provides a limited communication between components in a restricted manner. Although this style can be considered in limited parts of the system, in general it will not satisfy the needs and will restrict the system.

The predicted layered architectural style is very similar to the C2 style. The C2 style is a complex style appropriate for concurrent, heterogonous components distributed on different machines as stated by Taylor et al. (2010). The main motivation behind the C2 style is the need of component based development (Whitehead et al., 1995). Component based development is essential for prototype development where component reuse is critical when we consider the development of further versions of prototypes. The main parts of the C2 style are components and connectors. Components and connectors exist in a predefined top and bottom layered structure. The bottom of each component is connected to the top of the connectors. Communication is only allowed via connectors (Taylor et al., 1996). This limitation defined in C2 style should be broken in our architecture and additional connectors should be necessary between some of the components.

Architectural patterns are more specific application forms of the conceptual architectural styles. Architectural patterns are for reusable solutions for commonly seen problems in software architecture. The difference between architectural styles and patterns is not obvious in the literature and are generally mixed under the name of architectural styles and patterns. Apart from the C2 style, object-oriented and event-driven architectural styles and patterns are applied in the system.

Aside from architectural level, software design patterns are directly related to patterns on programming level as well. The architecture of the system should be able

to hide the details of the system by providing representative components. The best suitable software pattern for this purpose is the Façade Pattern.

The Invisible Eyes system is composed of two parts as described in the previous section. System component on the waist of the user is the main component where all the algorithms and logic is implemented. This component contains all the modules described in the previous section, namely RFID and Bluetooth modules, knowledge engine, algorithms, map related modules etc. Moreover, the system component on the hand of the user is for input and output, where the voice user interface is the basic module of the component. These two main components of the system are designed according to the Façade software pattern.

The Façade pattern (or facade pattern) is a software design pattern that provides a simplified interface to a larger body of code, such as a class library as described by Freeman (2004). A facade can make a software system easier to use, understand and test. The Facade design pattern is often used when a system is very complex or difficult to understand because the system has a large number of interdependent components. This pattern hides the complexities of the larger system and provides a simpler interface to the client. It typically involves a single wrapper class which contains a set of members required by the client. These members access the system and hide the implementation details. Figure 3 gives a very simple facade example. Client1 directly communicates with Facade without dealing with the complexity of the system. The complexity and interior details of the system is known by the Facade. Facade is a bridge between the complex system and the client. The client only deals with facade.

WaistFacade and *HandFacade* are the main components of the systems located on waist and hand of the user, respectively as seen in Figure 4. Facade components create a hierarchy between modules. Client components are able to use the facade components without dealing with internal details which are the responsibility of the facade components. Client components of the system are named *WaistClient* and *HandClient,* where the *WaistClient* is directly related with *WaistFacade* and the *HandClient* with *HandFacade*. The system logic is implemented in these client components.

The proposed system contains many modules and a hierarchy must be established among these modules. Therefore, facade components and client components using the facade components are necessary for the orchestration and synchronization of all the modules. Additionally, since the topic studied is state-of-

the-art strictly depending on the technological developments, new solutions and technologies may arrive later, which means the system maintenance and development are also of crucial importance. Facade pattern best fits the maintenance and development of the system. In the sustainability phase of the work, if additional modules are introduced, such modules can be easily integrated to the system because of the facade architecture.

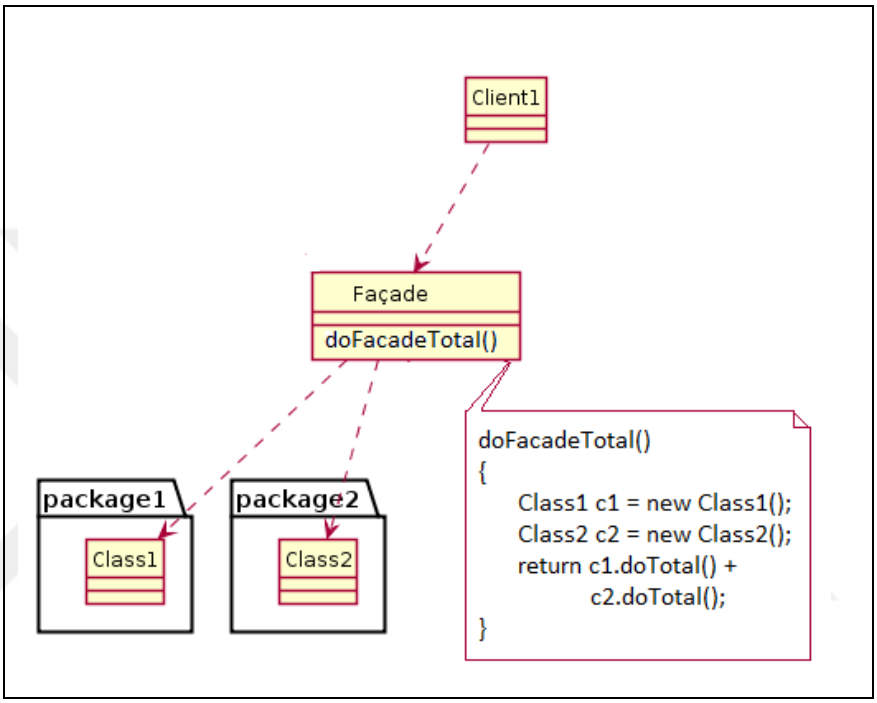


Figure 3: An example of Facade Pattern.

In the system, *WaistClient* mainly runs in three states as illustrated in Figure 5. Initially, the system is in idle state where the *WaistClient* listens for any instruction from the *HandClient*. A state transition from idle to evaluation state occurs when *HandClient* sends a command. In the evaluation state, *WaistClient* evaluates the path considering input data coming from *HandClient*. *WaistClient* executes all the system components using *WaistFacade*. After the evaluation state, system goes to monitoring state in which all the required data is sent to *HandClient*. Additionally, in the monitoring state, *WaistClient* continues to listen for instructions coming from *HandClient* and when needed, *WaistClient* may go to evaluation state for updated input data and then continue with monitoring state again. The system may change the state from monitoring to evaluation many times according to the user progress and needs of

the system. The *WaistClient* finally goes to idle state after finishing the jobs in the monitoring state and listens for new instructions.

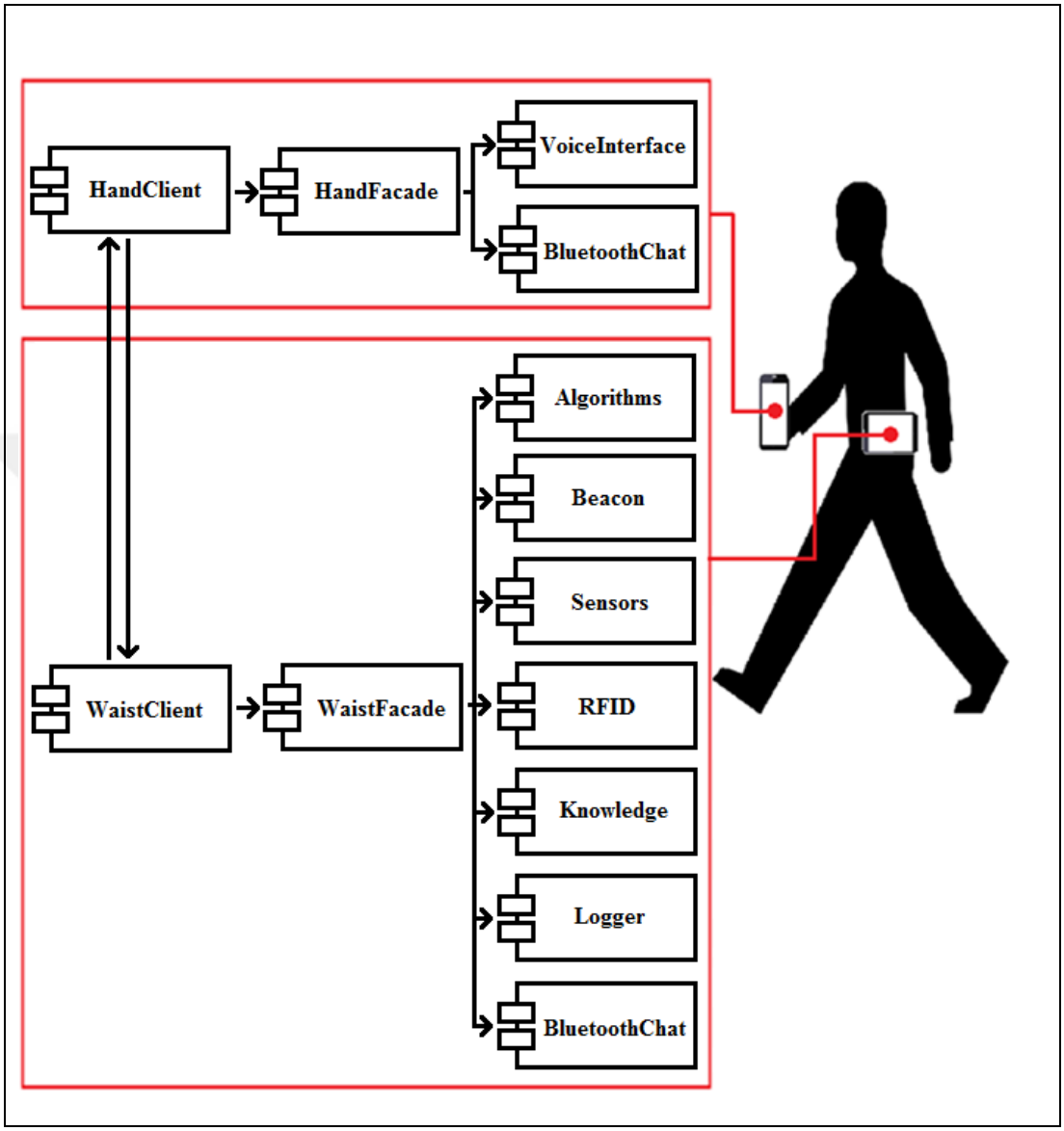


Figure 4: The general architecture of the system.

Furthermore, there is an internal mechanism in monitoring and evaluation states. For example, in the monitoring state, if the user progress does not fit with the system output, system goes to the error state which is inside the monitoring state. In the error state, system helps the user to achieve the given task by repeatedly making state transitions between monitoring and evaluation states.

The participatory design strategy is considered with the agile software development methodology as discussed in the previous section. Manifesto for agile

software development, the software must be developed by doing it and helping others to do it (Martin 2006). Additionally, working software is better than documentation and customer collaboration is critical. Therefore, we partially implemented and tested the design in each step.

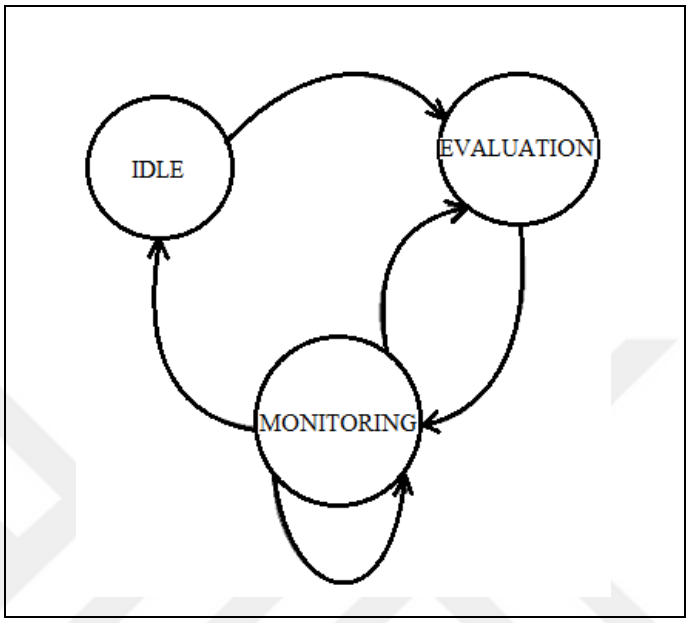


Figure 5: State diagram of the Waist Client System including the main states.

Initial tests were performed by the representative and the presented properties were said to be compatible with the user requirements. The initial design does not contain all the system components as described in the previous chapter. Only the RF reader module and related modules are implemented and tested on a simple path. In this implementation, shortest path module and RF reader module as well as graph representation and navigation commands were successful. However, since the RFID technology is a sensitive technology, some tags in narrow corridors were missed by the reader. This was because of the cancellation of the signals. Additionally, the same problem was observed on metal surfaces and on regions where metal content is dense.

Another critical point of the initial observations on the implementation was related to the communication channel between RF reader and mobile device. RF readers generally use the sound channel. In our architecture, this does not influence the operation. However, in the architecture depending on one mobile device this may be a problem because some operating systems like Android do not allow two devices to use the sound channel simultaneously.

Another observed point which will influence the usability of the system was that, user achieves more speed on second trials compared with the first attempt. User remembers the navigation commands and can move without waiting for the system to give output. Therefore, a training is required before using the system, which will inform the user about how to use the system.

3.5 Indoor Map Representation

Map of the building must be represented and included in the system to navigate user in indoor environment. The accessibility path specifications given in Chapter 2 must be correctly reflected in the system. The accessibility path is a graph for the system defined as an abstract data structure consisting of a finite set of vertices and edges. It is an undirected graph meaning both directions are possible. Nodes are predicted to be placed one meter away from each other where each node on the abstract data structure graph corresponds many RFID tags on the accessibility path. In one meter distance in the accessibility path, many RFID tags are placed depending on the tag type and range of the RF reader.

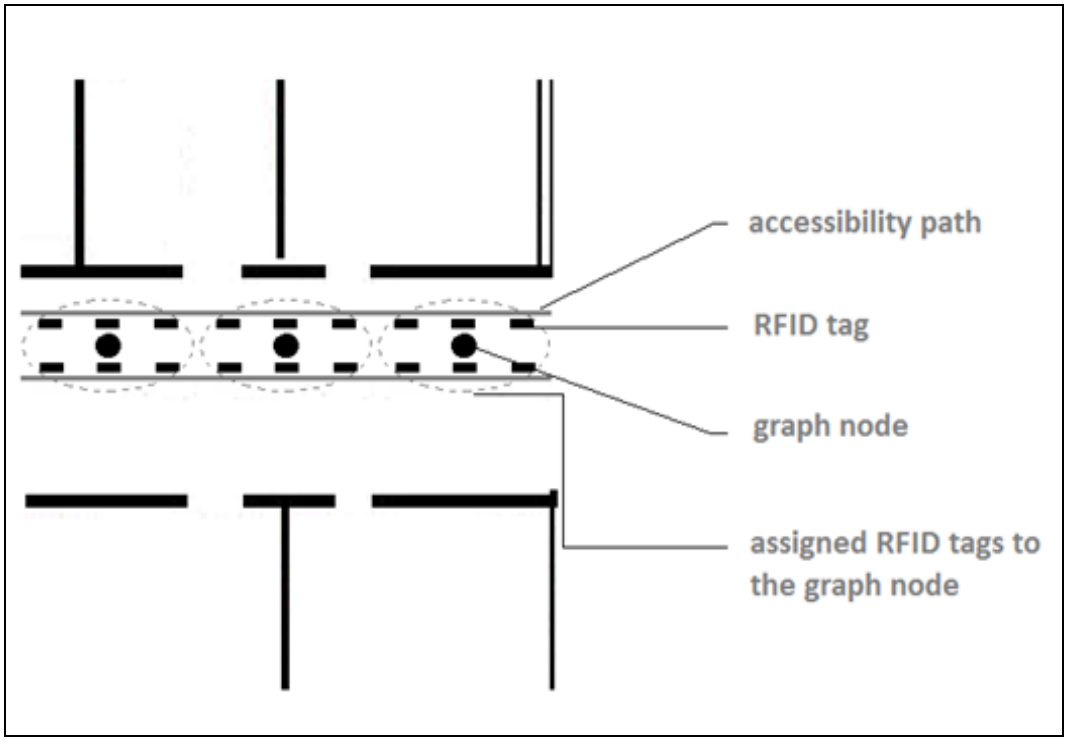


Figure 6: Accessibility path and its components.

In literature, hierarchical map structure is proposed for map representation. In the work of Lorenz et al. (2006) a hierarchical model is presented. Instead of hierarchically defining the map or graph, in our model, hierarchies are defined among components which are RFID tags and nodes of the graph. Two hierarchic layers for RFID system and similarly two hierarchic levels for Bluetooth positioning system are defined.

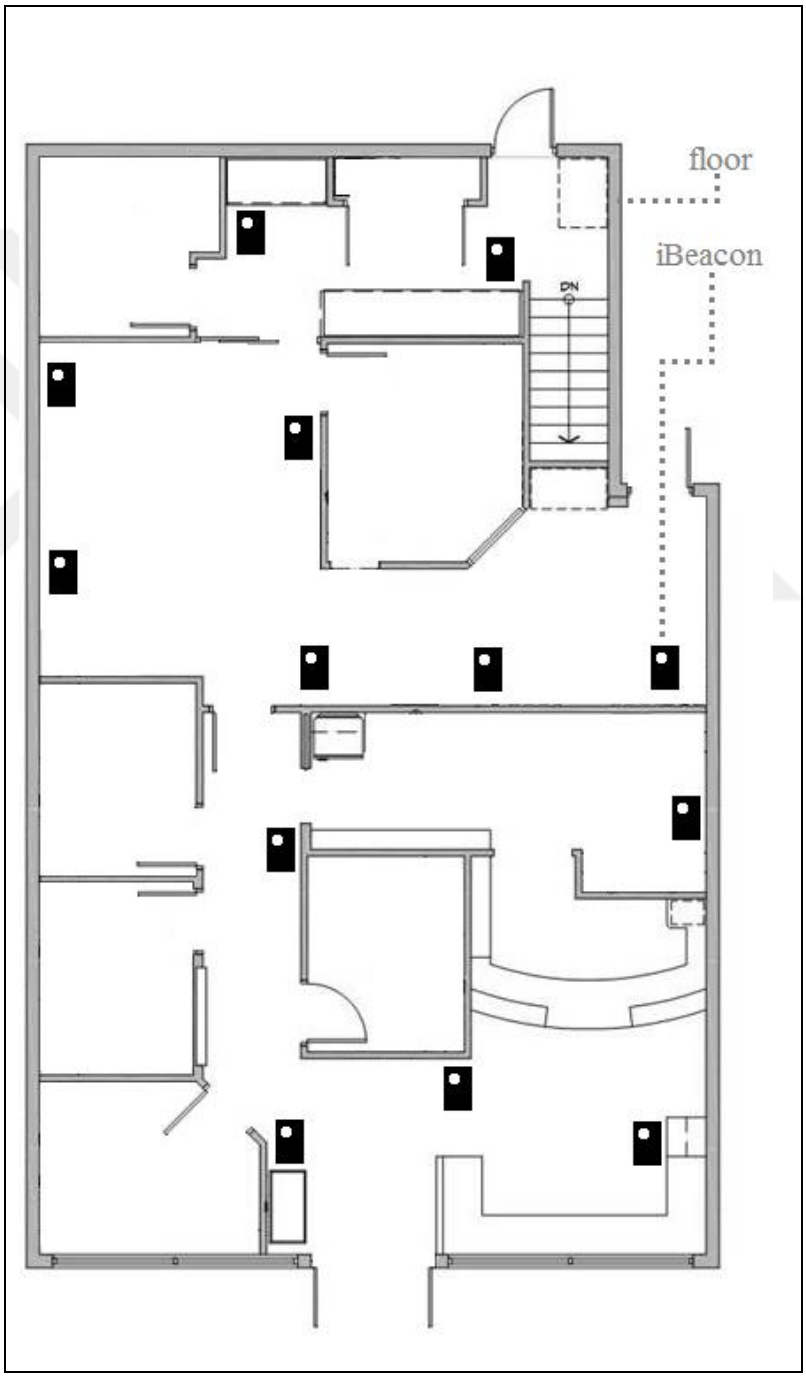


Figure 7: Bluetooth beacons are assigned to the unique positions on the floors.

In the RFID system, each passive RFID tag is assigned to a graph node and graph nodes are assigned to the positions on the floor as seen in Figure 6. Since the range of the RF reader is limited, these two levels are sufficient for RFID system. Each graph node contains six RFID tags as seen in Figure 6. This number is determined according to the initial trials and the number may change according to the distance between nodes, tag specifications, and reading range of the RF reader. A vote based classification is implemented and user is assigned to the winning graph node on the accessibility path.

Similarly, for Bluetooth positioning, each Bluetooth beacon is assigned to a specific position on the specific floor as seen in Figure 7. The system can identify the nearest Bluetooth beacons and predict the approximate distance of each identified Bluetooth beacon to the user. After these evaluations, the system can predict the position of the user related to the position of these surrounding Bluetooth beacons. The predicted positioning algorithm will be explained in the following sections.

Relative positions of graph nodes should be evaluated in clockwise directions and stored in the database. For example, if we consider five nodes as seen in the Figure 8, there are 16 possible navigations. If the starting point is Point 1, we may follow four possible steps: 1-2-3, 1-2-4, 1-2-5, 1-2-1. Here, the center point is Point 2 and we may evaluate the clockwise directions. For example, for the path 1-2-5, the required navigation command at Point 2 is “turn right on clockwise 3 direction”.

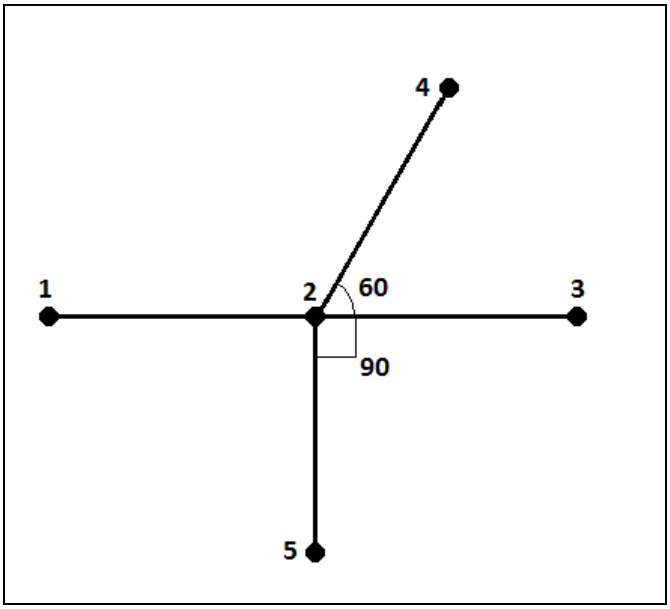


Figure 8: An illustrative graph representation.

Similarly, for the path 1-2-4, the required navigation command at Point 2 is "turn left on clockwise 10 direction". These clockwise directions will be evaluated and stored in the database for each point. This operation is complicated because of the number of possibilities but can be done automatically.

The required directions later will be queried from the database and converted into direction commands when needed. Another alternative can be defined as evaluating the required command on the fly, when the navigation instruction must be given to the user.

3.6 Intelligent Guidance

The most significant contribution of the work to the literature is the intelligent guidance and navigation where users input their purpose to the system and the system executes the given purpose using predefined rules to determine and order the relevant destination points. This idea is new and not considered in any study as far as we know. In this section the design principles of the intelligent guidance mechanism which converts the given purpose to the destination points, is explained.

Procedures and rules related with the operation of the building can be considered as information which is important for guidance of the users. These rules depend on the building and may show variety even between similar buildings. Therefore in our system, rules will be stored separately and the system will make it possible to include additional rules or change existing ones without any further modification to other parts of the system. A separate knowledge engine is implemented for this purpose. Each item in the rule is also associated with position information. An illustrative example can be given by considering these rules:

1. Cardiology Department has a Cash Point.
2. Cardiology Department has a Medical Admission Point.
3. Dr. Ahmet Yılmaz is a doctor in Cardiology Department.
4. Cardiology medical admission is the starting point in Cardiology Department.
5. In Cardiology department go to the Medical Admission Point before medical examination.
6. In Cardiology Department, go to the Cash Point before leaving.
7. In Cardiology Department the cost of Medical Examination is 100 TL.

8. Cardiology Department's Cash Point is assigned to graph node 138.
9. Dr. Ahmet Yılmaz room in the Cardiology Department is assigned to graph node 122.
10. Cardiology Department's Medical Admission Point is assigned to graph point 101.

Given such rules, if user gives a purpose such as "a medical examination to Dr. Ahmet Yılmaz in Cardiology Department" the system must guide the user considering the rules. A path will be determined as … → node 101 … → …node 122 →…→node 138 →… which means go to Medical Admission Point (node 101), then to Dr. Ahmet Yılmaz's room (122), and finally to the Cash Point (node 138). Additionally, the user can be informed during the navigation considering the rules such as "In Cardiology Department the cost of Medical Examination is 100 TL".

The system has a knowledge engine and destination generation mechanism to determine and order the destination points, given the purpose. The knowledge engine contains rules about the hospital as explained above. At the end of the evaluation, the purpose is converted to the rules and then these rules are converted to respective destination point. Furthermore, the system puts the destination points in order to minimize the total travel distance of the user.

Each destination has a corresponding node uniquely represented by node id, destination type, and destination information. Two destination types are defined in the system as *ordered* and *unordered*. The system cannot change the order of the ordered destinations when the precedence of destinations are determined and leaves them in place where they occur in the rule. On the other hand, the system is allowed to organize the unordered destinations. The situation is illustrated in Figure 9. The rule *R1* contains one ordered and six unordered destinations. The destination *D1.1* is an ordered rule in the first place of the chain. Accordingly, the system cannot change the place of *D1.1*. Nonetheless, the system can organize the destinations *D1.2* to *D1.6* to optimize the travel distance. Similarly, the rule *R2* has three ordered and seven unordered destinations. The destination *D2.1* is the first destination, the destination *D2.5* is the fifth one, and the destination *D2.10* is the last one. These places of the ordered destinations cannot be altered by the system. The system can alter the place of destinations between *D2.2* to *D2.4* and *D2.6* to *D2.9* but cannot change the sets. The unordered destinations between *D2.2* to *D2.4* lies between ordered destinations *D2.1*

and *D2.5*, so the system is not allowed to change it but can organize the unordered set to minimize the total travel distance as illustrated in Figure 9.

The problem of determining the order of the destinations in an optimum manner is similar to Traveling Salesman Problem (TSP). The system will organize the unordered nodes as described above by implementing the TSP. Details about the problem is included in the next section.

Additionally, Rule class also has a type property. Two rule types are defined as *tight* and *loose* types. These types are required for complex rules that contain other rules recursively. By default, each rule is tight. Rule types are similar to destination types.

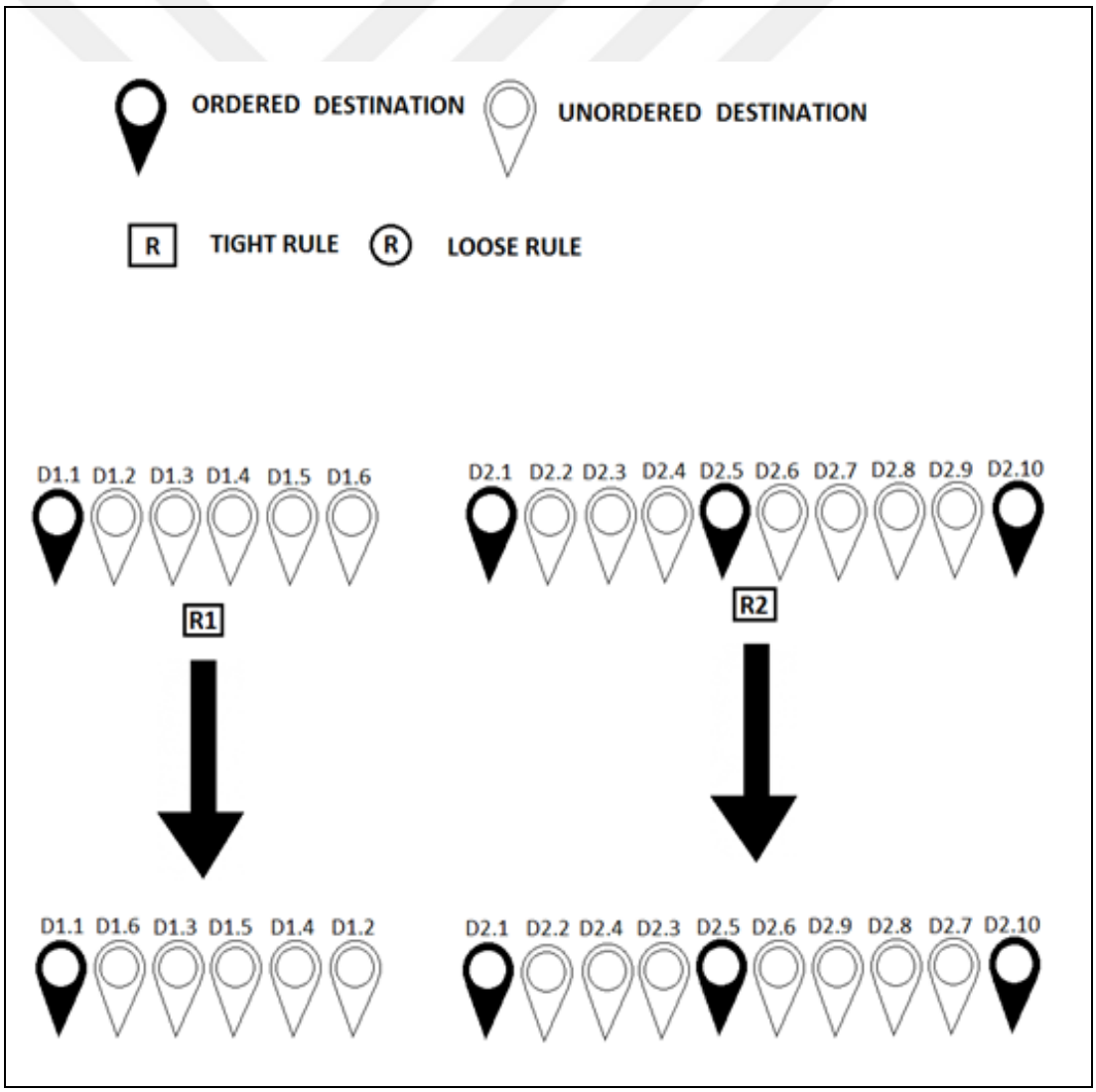


Figure 9: The difference between ordered and unordered destinations.

A tight rule means the place of the rule cannot be changed and the rule cannot be mixed with previous or following rules. On the other hand, loose rules can be mixed with each other to minimize the total travel distance, but the destination precedence for each rule again must be considered between destinations of the rule by the system. The case is illustrated in Figure 10.

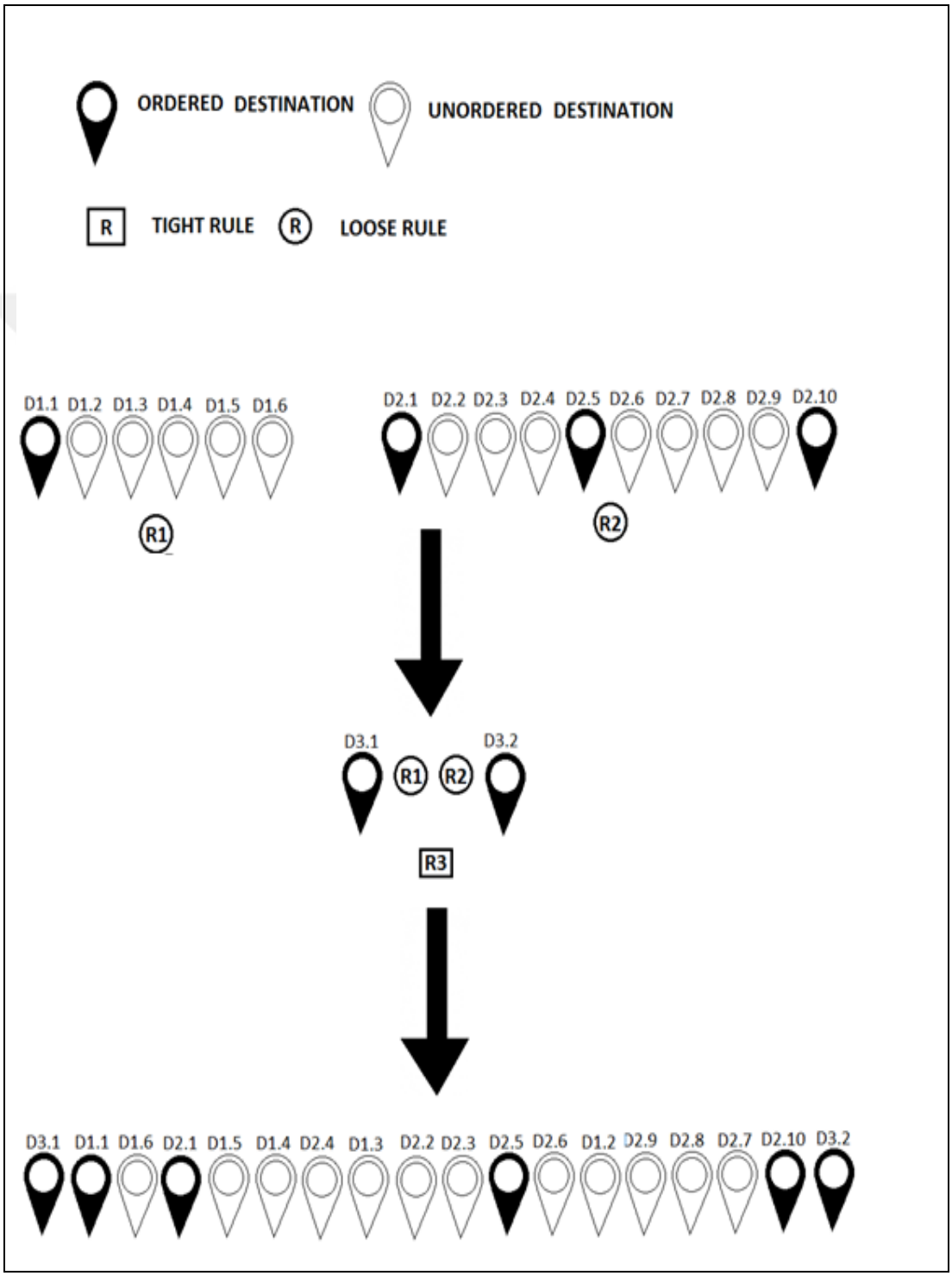


Figure 10: The difference between tight and loose rules.

Rule *R3* contains two ordered destinations and two loose rules. The destinations in two consecutive loose rules can be mixed. The system can organize the destinations between loose rules to optimize the travel distance as illustrated in the Figure 10. However, the precedence of the destinations as described in rule *R1* and *R2* must be obeyed. As seen in Figure 10, the system organizes the destinations between rules but still obeying the following precedence specified in rule *R1* and *R2*:

- unordered destinations *D1.2* to *D1.6* still comes after ordered destination *D1.1* as given in rule *R1*,
- unordered destinations *D2.2* to *D2.4* are still in between the ordered destinations *D2.1* and *D2.5* as given in rule *R2*,
- unordered destinations *D2.6* to *D2.9* are still in between the ordered destinations *D2.5* and *D2.10* as given in rule *R2*,
- the ordered destination *D2.1* still comes before *D2.5* and *D2.5* still comes before *D2.10* as given in rule *R2*,
- the ordered destination *D3.1* is the first rule and *D3.2* is the last one as given in rule *R3*.

This situation is very seldom but the proposed system is still able to cope with such rules. In this case, the problem becomes the Precedence Constrained Traveling Salesman Problem (PCTSP) which is a special form of the TSP and explained in the next sections. The PCTSP is not implemented but suggested as a future work in this study.

3.7 Traveling Salesman Problem

The Traveling Salesman (TSP), can be defined as a salesman starting a journey from a city, visiting each city in his list exactly once and then returning to the first city where he started. The salesman can choose the order of the cities to visit such that the total travel distance is minimized (Lawler, 1985). The problem is an old problem having a long history documented as early as 1759 by Euler.

Algorithms have been developed by researchers for TSP. For example, it is possible to formulate the TSP with linear programming (Dantzing, 1954) which is an optimization method where requirements are represented by linear relationships. In addition, the simplex method is a linear programming method which is suitable for problems involving a large number of constraints and can be adaptable for computers.

Furthermore, the cutting-plane method which depends on simplex algorithm, can be implemented for TSP (Applegate, 2003).

Because of the immense number of possible tours, it is not easy to obtain an optimal solution for TSP although the problem is conceptually simple. TSP is a special problem for computer science because the problem is an NP-hard problem. The complexity of the problem is factorial.

Apart from linear programming solutions to the TSP, other techniques are also designed. The Held-Karp algorithm is a dynamic programming algorithm designed for TSP (Held, 1962).

Also, the nearest neighbors (NN) algorithm is the one of the heuristic methods that can be applied to TSP (AlSalibi, 2013). In our work, the NN algorithm is chosen for the implementation of the TSP.

Several variations of the TSP are defined in the literature as symmetric traveling salesman problem (STSP), asymmetric traveling salesman problem (ATSP), multi traveling salesman problem (MTSP), and precedence constrained TSP (PCTSP) are some of the variations. PCTSP is mentioned in the next section.

The TSP is an old and well known problem having many variations and suggested solutions. In this work, the TSP is treated by NN algorithm considering the simplicity and efficiency of the method.

3.8 Precedence Constrained Traveling Salesman Problem

The PCTSP is a special form of the TSP where predefined constraints are included. The problem formally can be defined as find a tour from the first node to the node *n* of minimal length which takes given precedence constraints into account given a set of *n* nodes, distance for each pair of node, and precedence list. This variation can be applied to many real life problems such as scheduling, routing decisions, process sequencing, etc. Even though PCTSP is more suitable for real life applications, it is not investigated as deeply as TSP (Savelsbergh, 1995).

PCTSP is much more difficult than the TSP. Formulation of the model is not easy and so developing the algorithm and implementation is complicated. A branch-and-bound technique is suggested for the solution of PCTSP (Kusiak, 1987). Heuristic models are also delivered and presented in literature (Renaud, 2000). The pickup and delivery TSP which can be formulated as PCTSP is considered by Renaud et al. by

using heuristics (Renaud, 2000). Dynamic programming is also used for the solution of TSP with time windows and precedence constraints (Mingozzi, 1997).

Natural computing approaches are also efficiently implemented for the problem. Moon C. et al. proposed an efficient genetic algorithm for PCTSP (Moon, 2002). In their approach they modeled the problem by network model. Traditional TSP solutions are not appropriate for PCTSP because the constraints cannot be modeled adequately. The proposed genetic algorithms method shows superior performance compared to traditional alternatives. The disadvantage of genetic algorithms arises because of the randomness which is tried to be eliminated by Sung et al. and they worked on an adaptive evolutionary algorithm for PCTSP (Sung, 2014). Gambardella et al. (2012) worked on an ant colony optimization method which was able to solve many previously unsolved instances of the problem.

It can be concluded from the literature that the PCTSP is studied firstly by traditional optimization methods and algorithms. However, results obtained from the use of genetic algorithms are very successful and so genetic algorithms can be said to be efficient algorithms for PCTSP. Therefore, genetic algorithms are suggested to be used for complex rules in the study. The complex rules including many loose rules are introduced but not implemented in this study. In the target indoor environment, it is not possible to generate such complex rules and so this type of rules are not involved in the study. However, the formalization of the complex rules including loose rules recursively, is given. Additionally, in such cases if we want to put the corresponding destination points in order the problem becomes as a PCTSP instead of TSP. Therefore as described above even in this case the problem is solvable and genetic algorithms can be implemented as an efficient algorithm for the PCTSP. It will be an interesting future work to apply genetic algorithms for sorting the destination points in complex rules including loose rules recursively.

3.9 RFID and Bluetooth Indoor Positioning

In the system, the accessibility path contains RFID tags and a reader is placed close to the end of the white cane as explained before. The reader must be small in size and self-sufficient. Therefore, the reader must include a battery inside its compact size. The reading range of the reader is important. The reading range depends on type of

tag, type of reader, and frequency. Ultra high frequency (UHF) is required for long range identification considering a compact reader and cheap passive tags.

The width of the path is defined as constraints and requirements, defined in the previous sections. Each 1m piece of the accessibility path is considered as a node on the graph. The number of RFID tags on each piece depends on the type of the tag. Antenna design is not planned in this design and the suitable tag type should be found by testing and comparing the performance of the most common commercial types. The environmental conditions are also important for the performance of the RFID system. Tags are attached on the ground which decreases the reading range of the reader, due to the signal cancelation. Therefore, the reading range of the reader must be chosen so that it will compete with signal cancelation.

Along with the RFID position estimation, Bluetooth position estimation is planned in the study. RFID and Bluetooth methods can be combined as a hybrid method to best fit the requirements of the work. Bluetooth position estimation becomes crucial especially when the user is not on the walking path. In such cases, Bluetooth position system will evaluate the estimated location of the user and the system will guide the user through the nearest point on the walking path. In the study, neural networks and lateration methods are chosen and implemented for Bluetooth position estimation. Although this chapter is devoted to analysis and design, some design decisions must be validated before the implementation stage. Proof of concept and proof of the technology are two main types of validating the software architecture (Gorton, 2006). Bluetooth indoor positioning algorithm is crucial and related to the software architecture because the choice of the algorithm may change the predicted middleware and frameworks in the architecture. Therefore, before the implementation stage, the two chosen algorithms are tested by utilizing proof of concept and proof of technology. Proof of concept can be done by critical reviews and inspections. However, for proof of technology, partial implementation of the considered part of the system is required.

Neural networks can be used for indoor position estimation where indoor environment are divided into grids (G1 and G2) and Bluetooth transmitters are placed in known locations. A neural network can be trained with measured Received Signal Strength Indication (RSSI) values on each grid. Apart from RSSI values, some other measurable data can be used to increase the accuracy of positioning. In the work of Marco (2010), user orientation is also considered for neural network positioning.

In our work on neural network positioning with Bluetooth beacons, we could not obtained successful results. Initially we used RSSI and user orientation but the results were not found to be satisfactory. Later, to improve the accuracy we converted RSSI to distance (meters) and used these data for neural network positioning, but again could not reach a satisfactory level. Other than neural networks, HMM (Hidden Markov Models) can also be invoked for positioning as shown by Lees-Miller (2014) and Troels (2012). However, measurements of Bluetooth are not sensitive and stable which causes great errors in these techniques. Test results can be seen in Table 1. As the test environment, an 8m x 5m room is used.

Table 1: Neural network test results.

G1 value (m)	**G2 value (m)**	**Neural Network result**	**Expected result**	**Error**
3.86	9.52	[0.65 0.39]	[1 0]	NO
0.34	9.52	[0.65 0.36]	[1 0]	NO
5.83	13.82	[0.66 0.44]	[1 0]	NO
2.24	9.52	[0.65 0.35]	[1 0]	NO
4.28	12.61	[0.65 0.35]	[1 0]	NO
1.28	13.82	[0.65 0.36]	[1 0]	NO
4.76	4.75	[0.61 0.40]	[1 0]	NO
7.12	21.52	[0.67 0.41]	[1 0]	NO
7.12	1.15	[0.61 0.41]	[0 1]	YES
10.47	0.71	[0.61 0.42]	[0 1]	YES
11.49	0.84	[0.61 0.42]	[0 1]	YES
10.46	2.00	[0.61 0.42]	[0 1]	YES
8.65	0.71	[0.61 0.42]	[0 1]	YES
21.51	6.45	[0.61 0.42]	[0 1]	YES
19.73	1.80	[0.61 0.42]	[0 1]	YES
15.4	4.28	[0.61 0.42]	[0 1]	YES

The environment is divided into two grids named G1 and G2. In the middle of 5m walls, two Bluetooth beacons are attached. Training data is collected for both grids and the neural network is trained with that training data. For simplicity, user orientation is not considered in these trials.

Neural network Bluetooth positioning can be improved by using filters and averaging the measurement data but Bluetooth beacon data is not stable, and this instability causes neural network technique to be unusable. Additionally, the neural network must be trained and when indoor environment setup changes, the neural network must be trained again with newly collected train data. This requirement was discussed with the participants according to participatory design principles and was not approved.

The insufficient results obtained from Bluetooth techniques forced us to work on lateration technique. Although the technique is well known, some modifications were needed and we developed our *Weighted Multi Lateration Algorithm*. The proposed algorithm is implemented and tested according to the proof of technology principle of software architecture validation.

With perfect information, lateration method will give an exact, unique answer, the single point at the perfect intersection of the circles. With imperfect information, the circles will not intersect at a single point, in fact the circles may not intersect at all. In this situation, an estimate of the position is found by looking for the point that simultaneously minimizes the distance to all circles, for example by using mathematical techniques such as Least Square Estimation (LSE) as described in Cook (2005).

In our work, it is seen that the proposed *Weighted Multi Lateration Algorithm* can be used with imperfect information when circles do not intersect. Additionally, the algorithm considers the distance and finds a representative intersection point which is near to the closest points. The algorithm is named weighted lateration and finds the weighted centre. The algorithm described in Figure 11 is for at most three circles but can easily be extended for n circles.

```
WeightedLateration(Vertices)
SortVertices(Vertices)
if Vertices.length = 0
  then  return null
else if Vertices.length = 1
  then return null
else if Vertices.length = 2
   then  return FindIntersectionGiven2Vertices(Vertices)
Else if Vertices.length = 3
     then  return FindIntersectionGiven3Vertices(Vertices)
Else
     Vertices3Nearest[0] ← Vertices[0]
     Vertices3Nearest[1] ← Vertices[1]
     Vertices3Nearest[2] ← Vertices[2]
     return FindIntersectionGiven3Vertices(Vertices3Nearest)

FindIntersectionGiven2Vertices(Vertices)
       find point k1 and k2 // points intersecting with the line passing through center points v1
     and v2 of circles and the circles
          Ratio1 ← v1.RSSI / (v1.RSSI + v2.RSSI)
          Ratio2 ← v2.RSSI / (v1.RSSI + v2.RSSI)
          p.X ← (k1.X*Ratio2 + k2.X*Ratio1) / (Ratio1 + Ratio2)
          p.Y ← (k1.Y*Ratio2 + k2.Y*Ratio1) / (Ratio1 + Ratio2)

FindIntersectionGiven3Vertices(Vertices)
       find point p1 and p2 // points intersecting with the line passing through center points v1
     and v2 of circles and the circles
          Ratio1 ← v1.RSSI / (v1.RSSI + v2.RSSI)
          Ratio2 ← v2.RSSI / (v1.RSSI + v2.RSSI)
          pp1.X ← (p1.X*Ratio2 + p2.X*Ratio1) / (Ratio1 + Ratio2)
          pp1.Y ← (p1.Y*Ratio2 + p2.Y*Ratio1) / (Ratio1 + Ratio2)
       find point p3 and p4 // points intersecting with the line passing through center points v1
     and v3 of circles and the circles
          Ratio3 ← v1.RSSI / (v1.RSSI + v3.RSSI)
          Ratio4 ← v3.RSSI / (v1.RSSI + v3.RSSI)
          pp2.X ← (p3.X*Ratio2 + p4.X*Ratio1) / (Ratio3 + Ratio4)
          pp2.Y ← (p3.Y*Ratio2 + p4.Y*Ratio1) / (Ratio3 + Ratio4)
       find point p5 and p6 // points intersecting with the line passing through center points v2
     and v3 of circles and the circles
          Ratio5 ← v2.RSSI / (v2.RSSI + v3.RSSI)
          Ratio6 ← v3.RSSI / (v2.RSSI + v3.RSSI)
          pp3.X ← (p5.X*Ratio2 + p6.X*Ratio1) / (Ratio5 + Ratio6)
          pp3.Y ← (p5.Y*Ratio2 + p6.Y*Ratio1) / (Ratio5 + Ratio6)
         Ratio7 ← v2.RSSI / (v2.RSSI + v3.RSSI)
          Ratio8 ← v3.RSSI / (v2.RSSI + v3.RSSI)
          Ratio9 ← v1.RSSI / (v1.RSSI + (v2.RSSI + v3.RSSI) / 2)
          Ratio10 ← v2.RSSI / (v1.RSSI + (v2.RSSI + v3.RSSI) / 2)
          ppp1.X = (pp1.X*Ratio8 + pp2.X*Ratio7) / (Ratio7 + Ratio8)
          ppp1.Y = (pp1.Y*Ratio8 + pp2.Y*Ratio7) / (Ratio7 + Ratio8)
          ppp2.X = (ppp1.X*Ratio10 + pp3.X*Ratio9) / (Ratio9 + Ratio10)
          ppp2.Y = (ppp1.Y*Ratio10 + pp3.Y*Ratio9) / (Ratio9 + Ratio10)
          rp.X = ppp2.X
          rp.Y = ppp2.Y
          return rp
```

Figure 11: The weighted lateration algorithm. The algorithm is for at most three points and can be extended for *n* points.

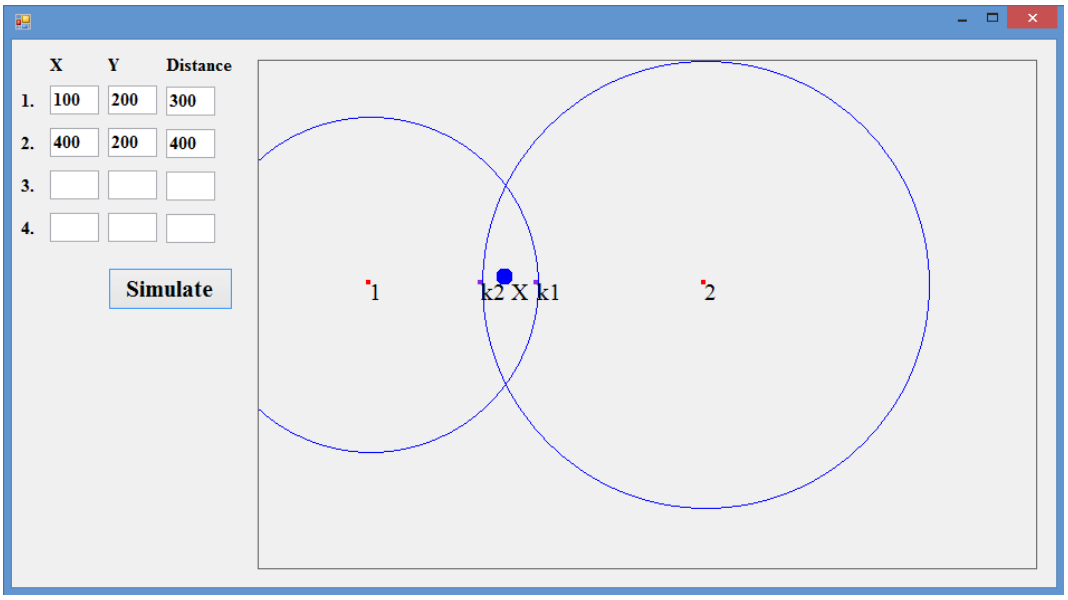


Figure 12: Two points with intersecting circles.

The validation of the *Weighted Multi Lateration Algorithm* is performed by using Visual Studio software development environment and C# programming language. Predefined cases are artificially constructed. These cases are special cases where the common lateration algorithm is not sufficient. Circle in the graph represents the evaluated distance of the Bluetooth beacon to the reader from the strength of the signal. Therefore, diameter of the circle represents the estimated distance of the Bluetooth beacon and closer the Bluetooth beacon, the smaller the diameter of the circle becomes.

The proposed *Weighted Multi Lateration Algorithm* is simulated considering these imperfect cases. These cases include disjoint circles, two, three and four points. Figure 12 – 19 are the simulation results obtained from these cases. As seen from the figures, the proposed algorithm was able to obtain a prediction for the user position in all the artificial imperfect cases.

In Figure 12, the case in which two Bluetooth beacons with intersecting signal strength circle is simulated. As seen from the figure, the algorithm tries to determine the position of the user according to these data. Firstly, the algorithm finds the point k_1 and k_2, which are points intersecting with the line passing through center points, v_1 and v_2, of circles. Then the signal strength ratios are evaluated. Finally, the algorithm finds the weighted center of the line between points k_1 and k_2. This point, represented as a bold dot in the Figure 12, is the approximate position of the user predicted by the algorithm.

The cases for perfectly and pairwise intersecting three Bluetooth beacons are given in the Figures 13 and 14, respectively. The algorithm, predicts the weighted center as the approximate position of the user in these cases similarly for the previous case. The only difference is that in this case, the algorithm finds the center point of a triangle instead of a line. This triangle is obtained as the intersection region of lines which are crossing the centers of the circles considering the strength of the signals coming from the Bluetooth beacons.

The algorithm considers the nearest three Bluetooth beacons when the number of perceived Bluetooth beacons is greater than three. This idea of considering the nearest three Bluetooth beacons can be extended to more than three. In such a case, the weighted center of the intersection of pairwise lines crossing the related centers of the circles, will be the center of a polygon.

The number of edges will be related to the number of Bluetooth beacons considered by the algorithm. For example, if the algorithm is modified to consider four nearest Bluetooth beacons, then the polygon will be quadrilateral and the weighted center of this quadrilateral will be the estimated position of the user.

In Figure 15, the case is simulated. There are four Bluetooth beacons recognized by the system as seen in the figure. The system considers the nearest three of these four Bluetooth beacons which are point 1, point 2 and point 3 in order. Point 4 is the farthest Bluetooth beacon and therefore is eliminated by the system.

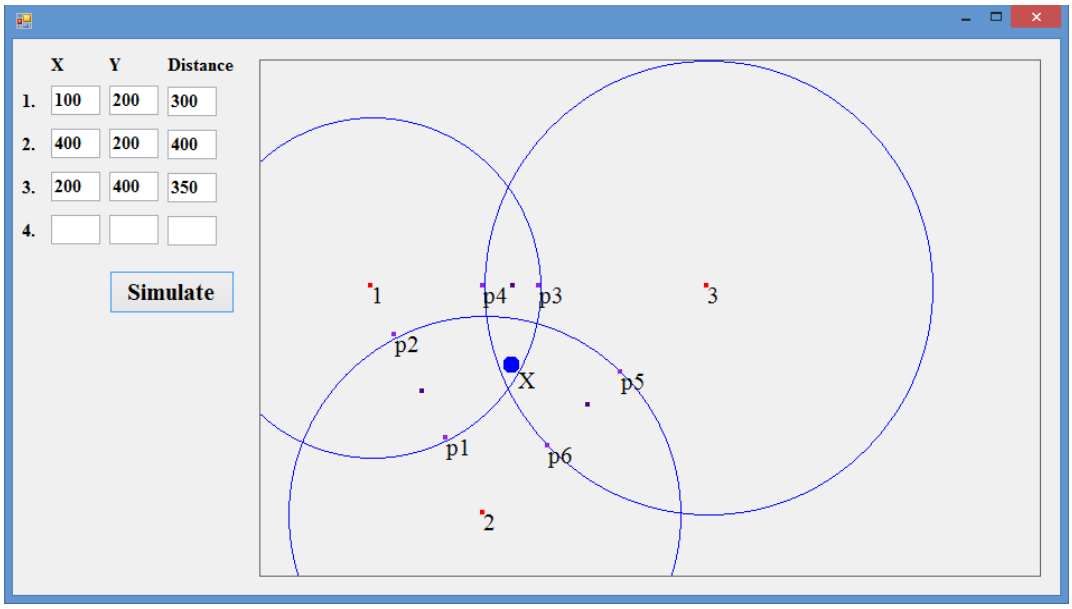


Figure 13: Three points with perfectly intersecting circles.

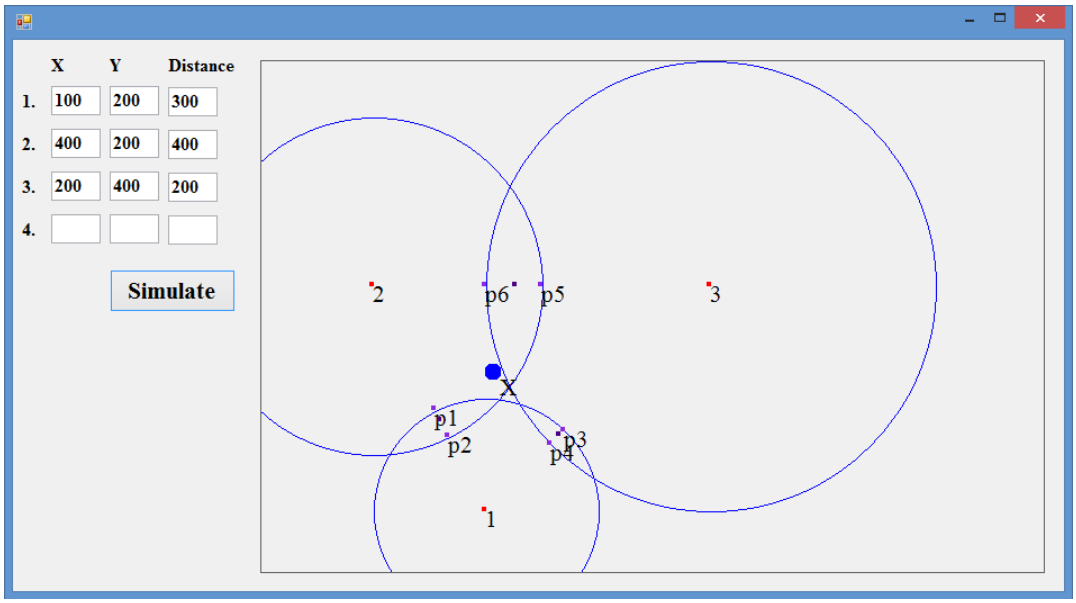


Figure 14: Three points with pairwise intersecting circles.

The algorithm cannot find a center point if only one Bluetooth beacon is recognized as illustrated in the simulation seen in Figure 16. In this case, the position of the user can be estimated as inside the area which is constructed by the circle centered at the position of the recognized Bluetooth beacon with diameter evaluated according to the signal strength.

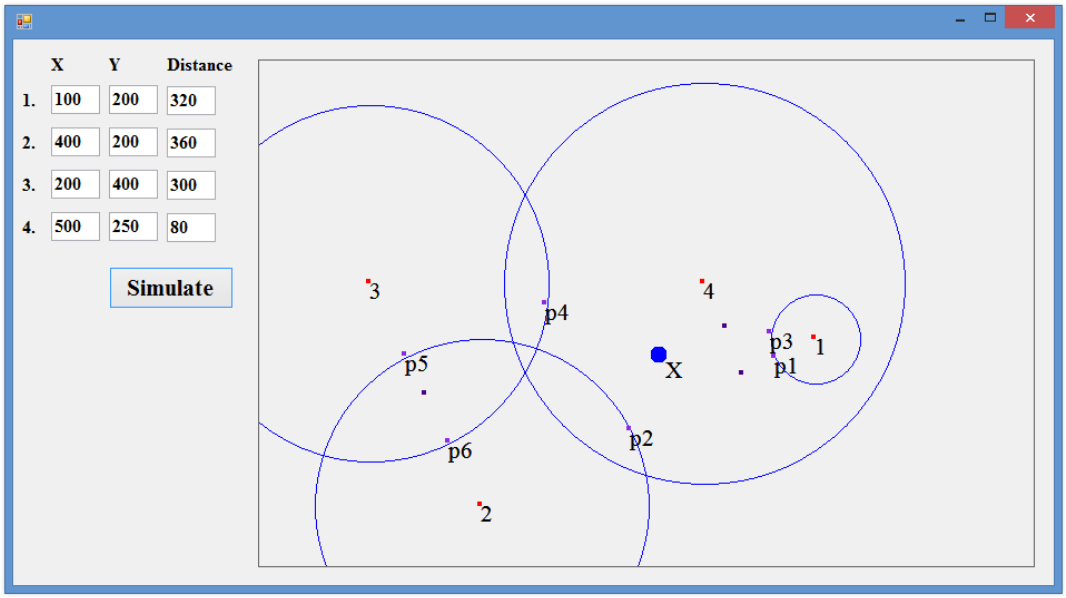


Figure 15: Four points with partially intersecting circles. Point 4 is omitted according to the algorithm. Point 4 represents the furthest Bluetooth beacon.

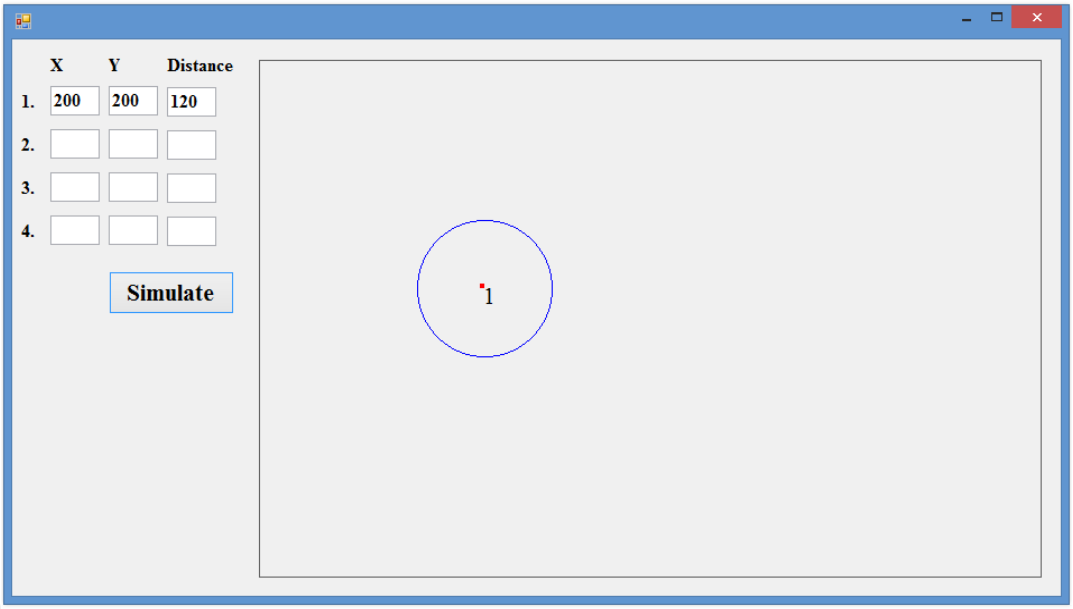


Figure 16: One point. Represents the case in which system recognizes only one Bluetooth beacon.

The advantage of the algorithm is to produce a result when the circles are disjoint. Since the algorithm is not designed to find the joint areas, in these cases again it will be able to produce the output. This case is simulated in Figure 17-19.

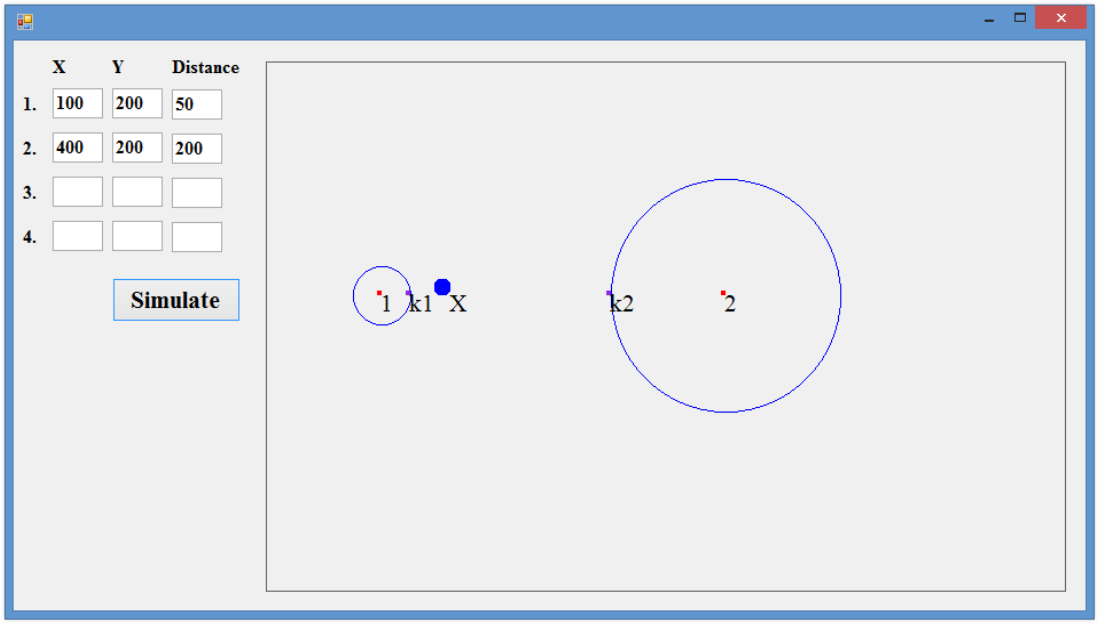


Figure 17: Two points with disjoint circles.

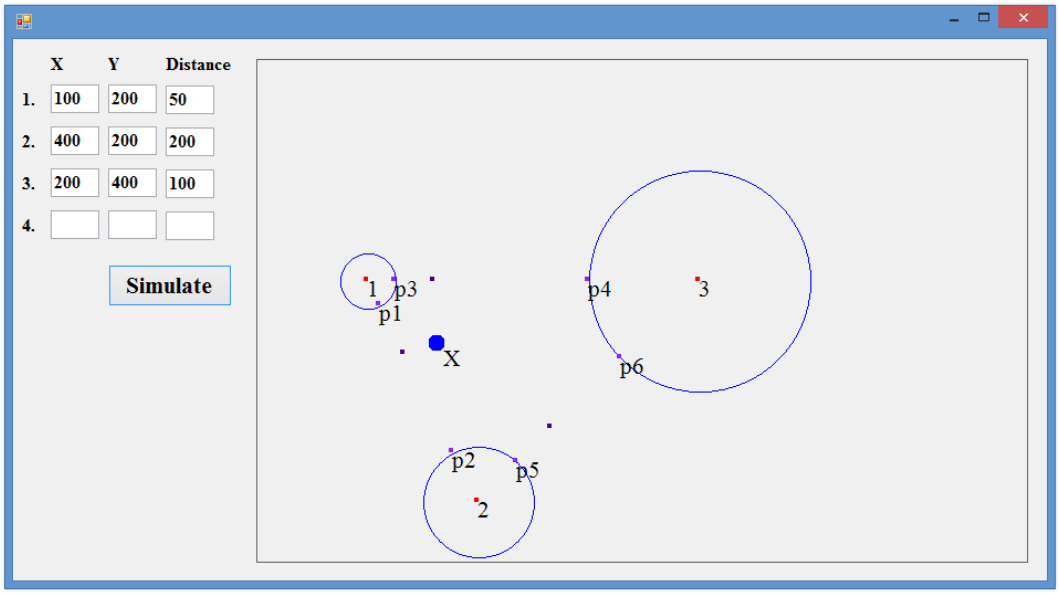


Figure 18: Three points with disjoint circles.

In Figure 17, the case of recognizing two Bluetooth beacons having disjoint circles is simulated. The system finds the weighted center point in the line between two center points and this point is the bold dot in the figure. Similarly, the case for three and four disjoint Bluetooth beacons are simulated and corresponding center points are evaluated by the algorithm. Results for the case of three and four Bluetooth beacons having disjoint circles, can be seen in Figure 18 and Figure 19, respectively.

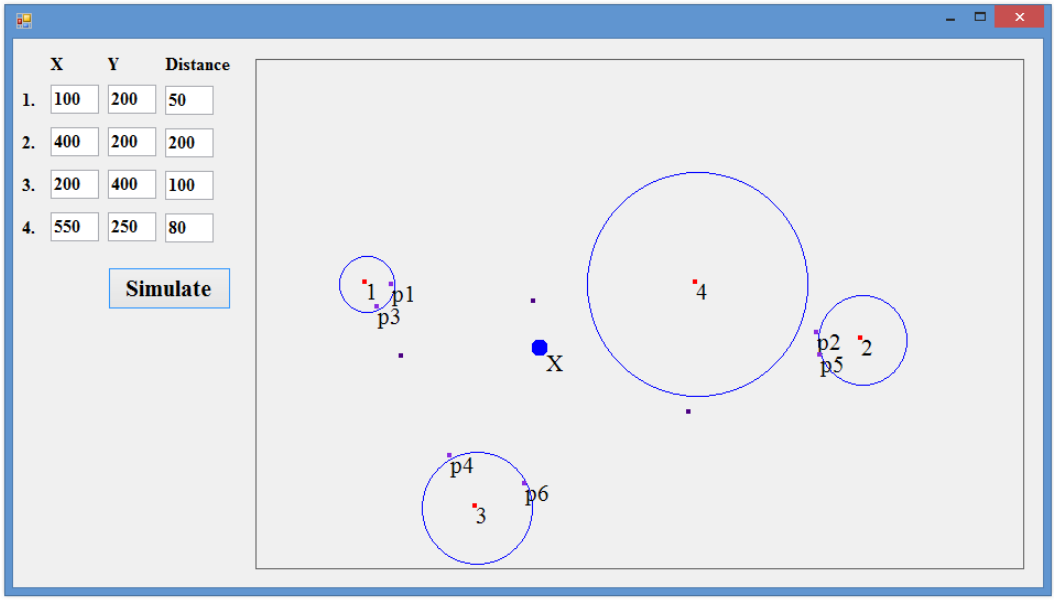


Figure 19: Four points with disjoint circles. Point 4 is omitted according to the algorithm. Point 4 represents the furthest Bluetooth beacon.

3.10 Concluding Remarks

The analysis and design of the study is given in this section. The analysis includes determining constraints and requirements with the help of participatory design team taking the related rules and regulations into consideration. The design, involves the design of main components, software architecture, intelligent guidance and navigation mechanism, and related algorithms.

Software requirements analysis was the first stage of the work. In this stage a team including the stakeholders which are users, developers, specialists, and managers is constructed. The study is a special study that cannot be seen as a common software development process or as only a theoretical study. The study aims to make contribution to the life of visually impaired persons and also to the literature by introducing an intelligence guidance and navigation system. Therefore, participatory design strategy is followed in the study. According to the participatory design strategy, the intended special users who are visually impaired people in our case, must be involved especially in the design stage of the work. The contribution of the users is not limited with the design stage and they are a core element of every stage of the study. Additionally, the participatory design team is not limited to end users and also includes developers, specialists, and managers as mentioned previously. After the regular meetings and discussions with the participatory design the software requirement specifications and constraints are determined and listed in this chapter.

Software design artifacts which are system components, software architecture, and algorithms used in the system, are detailed in this chapter. The implementation details of these design elements will be given in the next chapter. The system is divided into two components considering the needs of the users and the system. The system must be able to evaluate the orientation of the user correctly and users need to be free of using the system without any extra motion limitation like holding the mobile device in a stable direction. Therefore, to achieve both of the requirements, we divided the system into two systems. The system will be deployed to the main system which is located at the waist of the user and the system user interface will be deployed on another mobile device located at the hand of the user.

Software architecture is also considered and the main architectural styles and patterns are designed. The system will be implemented according to a layered architecture. The hierarchy between layers will be considered without inclusion of any

strict rule. Additionally, the complexity of the components will overcome with the use of Façade and Singleton design patterns.

The main contribution of the study was the proposal of the intelligent guidance and navigation system for visually impaired people. For this part of the study, the destination generation mechanism and related algorithms are designed. Therefore, the TSP and PCTSP which is a special form of the TSP are mentioned with practical and efficient solution algorithms.

Apart from analysis and design, the system is partially implemented to validate the Bluetooth dependent positioning part of the work. For this validation, proof of methodology and technology which is a software architecture test and validation methodology is applied. The neural network is not sufficient and therefore not appropriate as the result of the testing and validation study. Therefore, a novel lateration algorithm named *Weighted Multi Lateration Algorithm* is designed. The algorithm is implemented and proof of the algorithm is performed. At the end of the simulations which cover the imperfect cases, the algorithm was able to predict the position of the user in each case.

The implementation details of the designed and described system will be explained in the next chapter. In each stage of the implementation, the partially achieved system will be reviewed and tested with the inclusion of participatory design team.

Chapter 4

SYSTEM IMPLEMENTATION

This chapter includes implementation details of Invisible Eyes. In the previous chapters, the design details, techniques, algorithms, as well as the contribution of the study to the literature has already been included. This part of the work aimed to include implementation details to complement the previous sections.

In the first part, software patterns applied in the design and implementation is explained, where the Beacon Component is given to illustrate the software patterns. Later, major views of the systems are presented starting from automata view, then the components of the system are visualized, and GUI of the system is described. The chapter ends with the explanation of the knowledge engine and destination generation mechanism of the system as well as implementation of the algorithms.

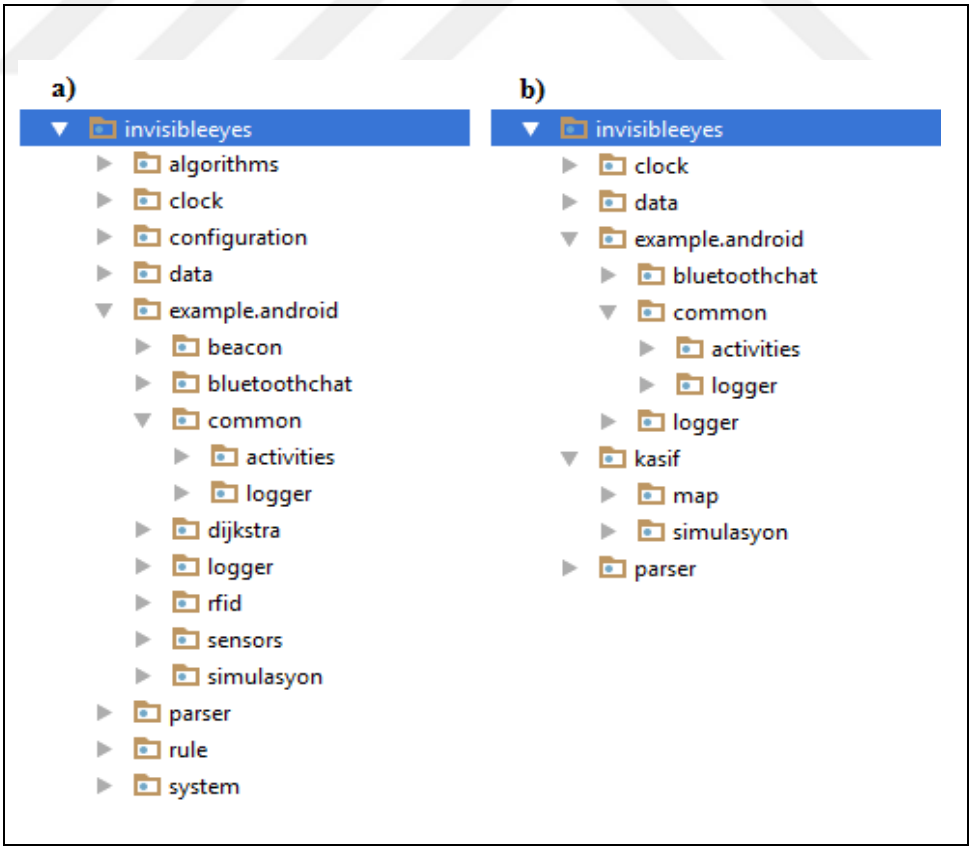


Figure 20: A general view of the packages in a) main systems and b) the system including GUI.

4.1 Implementation Environment and Hardware

The system was developed on the Android operating system in Java programming language, using Android Studio 1.4 which is one of the free integrated development environment (IDE). The main packages of main project and GUI project can be seen in Figure 20.

The IDE is installed on a home computer which has 2.50 GHz processor and 8 GB memory. The system is deployed into two mobile devices. The device holding the main project runs the Android 4.4.2 operating system, having 1.6 GHz processor, and 2 GB RAM. The other device is devoted to the second main component or project of the system which holds the user interface. This device has a 1.6 GHz processor and 1.5 GB memory with Android 4.2.2.

Table 2: The specifications of the RF reader.

Property	**Value**
Width	51 mm
Height	50 mm
Thickness	15 mm
Weight	35 g
Battery	Built-in rechargeable lithium-ion battery, 360mAh, Charging via Micro USB 5pin
Standby Time	About 17 hours
RFID Read Time	About 2 hours
RFID Reader Chip	PHYCHIPS PR9200
Interface	3.5mm TRRS(4-pole) CTIA Plug for Smart Phone Headset Jack
Operating Frequency	865.7 ~ 867.5 MHz
Operating System	Android 2.3.3 above
Antenna	Circularly Polarized Antenna
RFID Read Distance	About 1.0m(depend on tags)

A compact and self-sustainable ultra-high frequency RF reader is attached on the end of the white cane. The tags which are placed on the floor and specify the borders of the accessibility path, are recognized with the help of this RF reader. The identification number which uniquely identifies the tag recognized by RF reader is

delivered to the mobile device holding the main project. The specifications of the RF reader is given in Table 2.

A Bluetooth beacon is a broadcaster which transmits periodic advertising packets containing information used by the receiver (Gast, 2015). These packets are thrown out into the air and receivers catch these packets to process.

4.2 Software Patterns

In the system, as mentioned in the Analysis and Design chapter, singleton and façade patterns are used as design patterns. The Beacon component details are presented as an example of the use of the Façade design patterns. The *MainActivity* class seen in Figure 46 of Appendix A is for test purpose and represents Waist Facade. Figure 21 illustrates the relationship between objects. The operations and attributes of the objects are not given because the purpose of the figure is to describe the relationship between objects.

BeaconScanner is the responsible object for scanning and reading the beacon data. This object has the relation as composition with *BluetoothAdapter* which is an element of the Android library. *BeaconScanner* can send data coming from beacons to any object implementing the *BeaconListener* interface as seen in Figure 44 in Appendix A.

MainActivity has a beacon scanner as composition and while constructing the *BeaconScanner,* gives this object and the activity context via the constructor of *BeaconScanner* as seen in Figure 45 in Appendix A. In Android programming, Bluetooth libraries like other sensor libraries needs application or activity context, and that is why such a dependency between *MainActivity* and *BeaconScanner* is required. *IBeacon* seen in Figure 43 in Appendix A, is the data object encapsulating Bluetooth beacon data.

The main components of the system which are described in subsequent sections, are designed as singletons. The Singleton pattern restricts the instantiation of a class to one object. This is suitable when exactly one object is required to manage actions among the system. In the system, exactly one object is required for most of the components which makes extensive use of the singleton pattern.

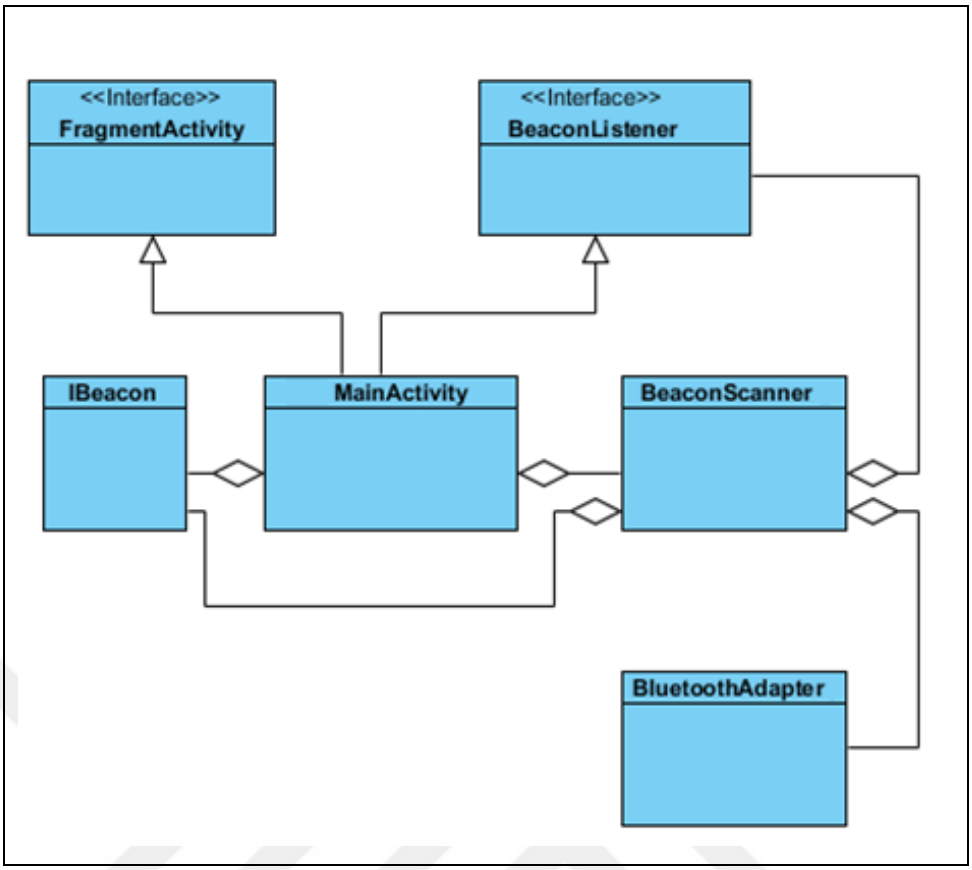


Figure 21: Architectural view of the beacon component.

4.3Automata View of the Main System

The system is governed by two main automata, one for system user interaction and one for navigation, named Interaction Automaton and Navigation Automaton, respectively.

Interaction automaton as described below is the intelligent part of the system. Destination points are identified and sorted by this automaton and for each destination point, interaction automaton executes navigation automaton for each destination.

4.3.1 Navigation Automaton

The navigation automaton as seen in Figure 22 contains seven states. Initially, the system is in *Idle* state when a state transition occurs, it forces the system to move to the *Evaluation* state. Then, the system starts to evaluate the shortest path and determine the required navigation commands such as “turn left to clockwise 9 direction”

depending on the destination point inputted by the Interaction Automaton. At the end of evaluation state, a state transition occurs and the current state changes to *Monitoring* state. In the *Monitoring* state, the orientation and position of the user is monitored and navigation instructions are given depending on the progress of the user.

In the *Monitoring* state, some undesirable conditions might appear as listed below:

- *The orientation (heading) of the user is not as required*: In this case, a state transition from Monitoring to *Correct Heading* state appears. The user is informed about the problem and the required instruction is given considering the current orientation of the user.
- *User is on a wrong node*: The accessibility path is represented as a graph containing nodes and edges between these nodes as described in the previous sections. The position of the user is traced by the system and the user is assigned to the nearest node. If the actual node is on the shortest path, *Progress Manager* is updated but if user goes out of the shortest path, a new shortest path is evaluated from the current node and the user is informed.
- *User is out of accessibility path*: In this case, the current position of the user is evaluated using Bluetooth beacons and user is navigated through the nearest point on the accessibility path.
- *User is out of accessibility path and position cannot be determined*: In this case, the user is informed and assistance is required.

4.3.2 Interaction Automaton

Interaction automaton containing three states is illustrated in Figure 23. Initially, it is in *Idle* state and when purpose is specified, state changes to *Evaluation* state. In the *Evaluation* state, the rule engine identifies and prioritizes the destinations depending on the predefined rules. The automaton moves to the *Waiting* state after evaluation finishes. In the *Waiting* state, for each destination, Navigation automaton is called and Interaction automaton waits until destination is reached. After all the destinations are completed, the Interaction automaton goes back to *Idle* state.

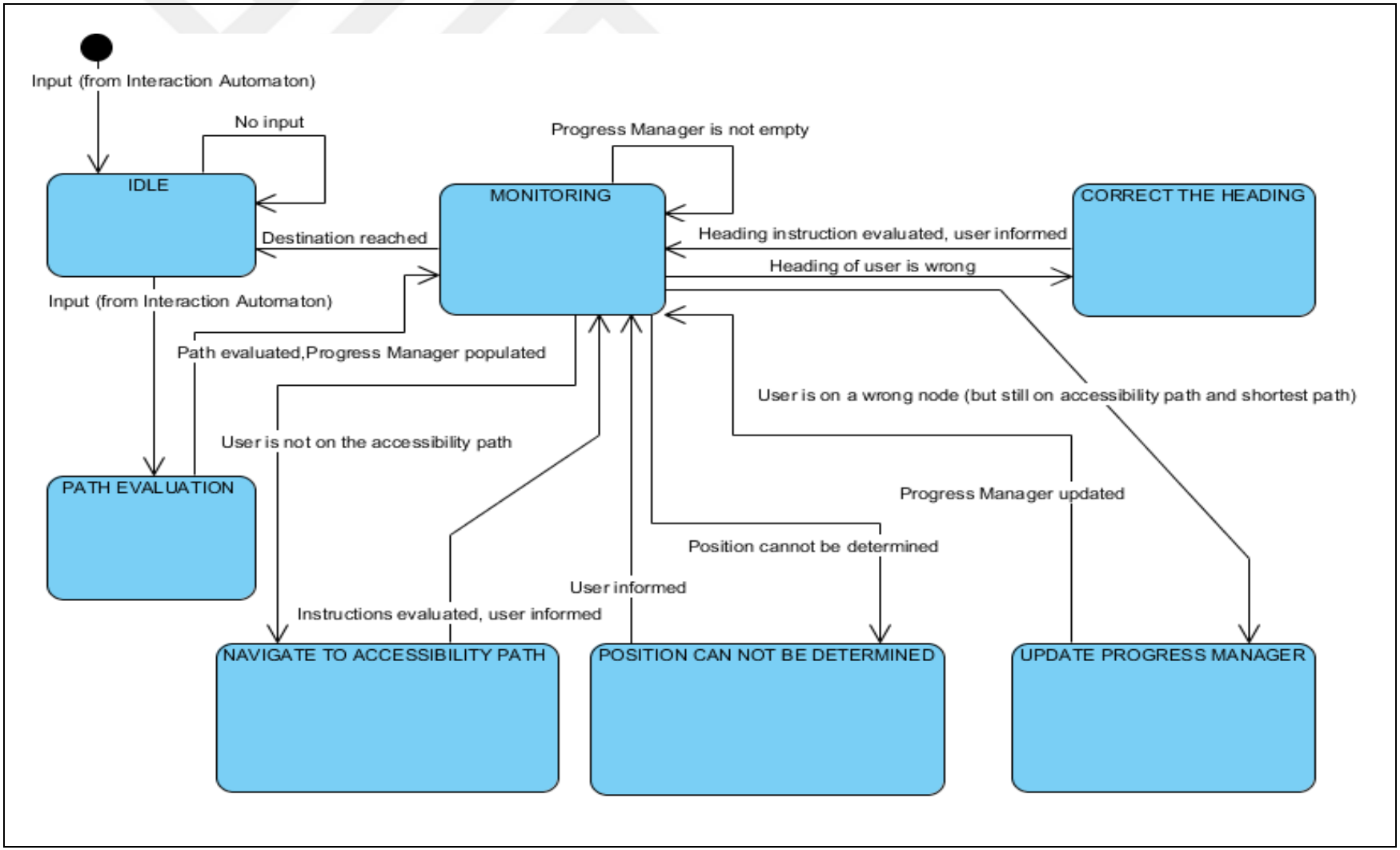


Figure 22: Navigation Automaton.

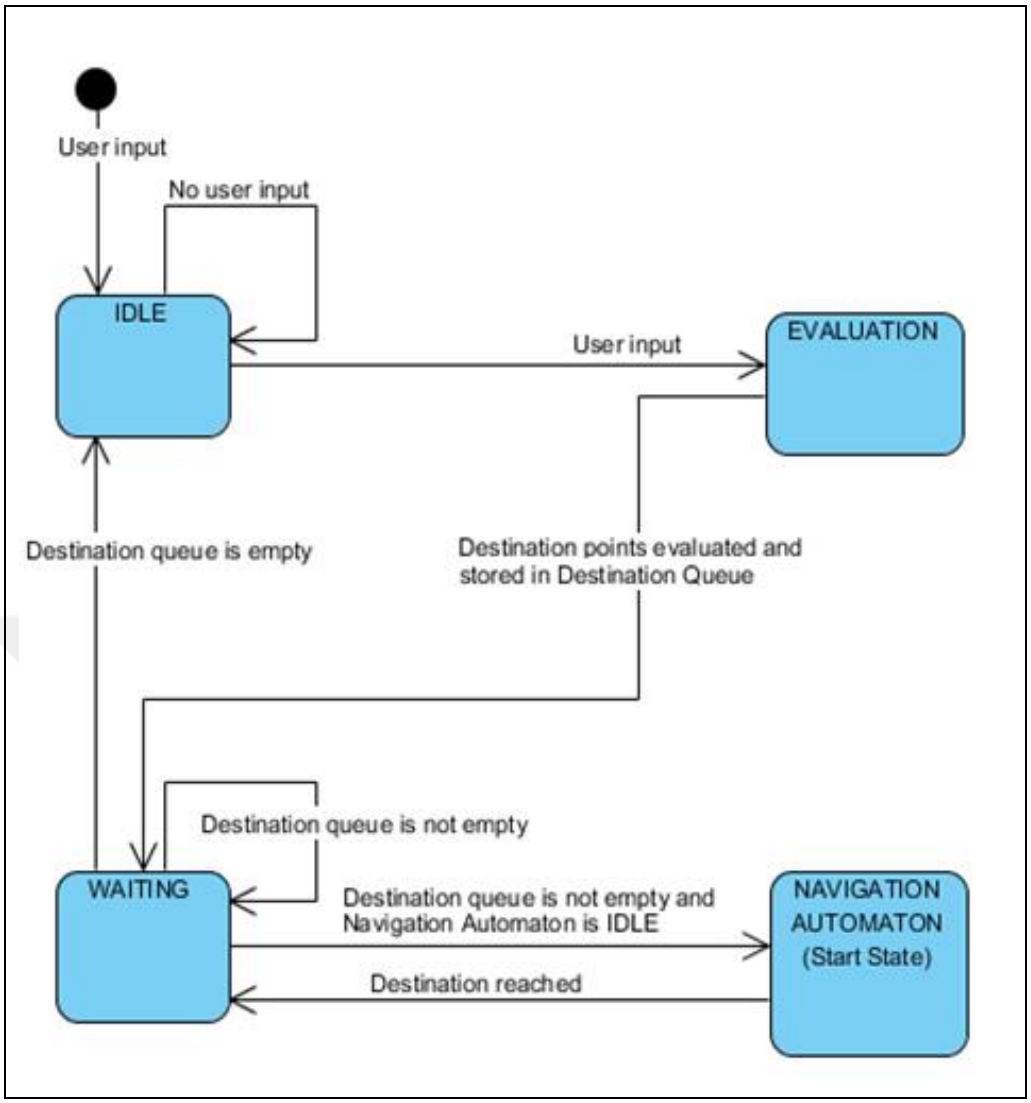


Figure 23. Interaction Automaton.

4.4 Component View of the System

The implemented system architecture is a layered architecture that is very similar to the C2 architectural style as seen in Figure 24 and designed in previous chapter. In C2 style, components and connectors are arranged in layers and the architecture has a defined top and bottom. The top of the architecture is the left hand side of the Figure 24. Components are aware of the other components that reside above them. Therefore, they may send requests and events that travel up the architecture. Components may also send out notifications, messages which travel down the architecture.

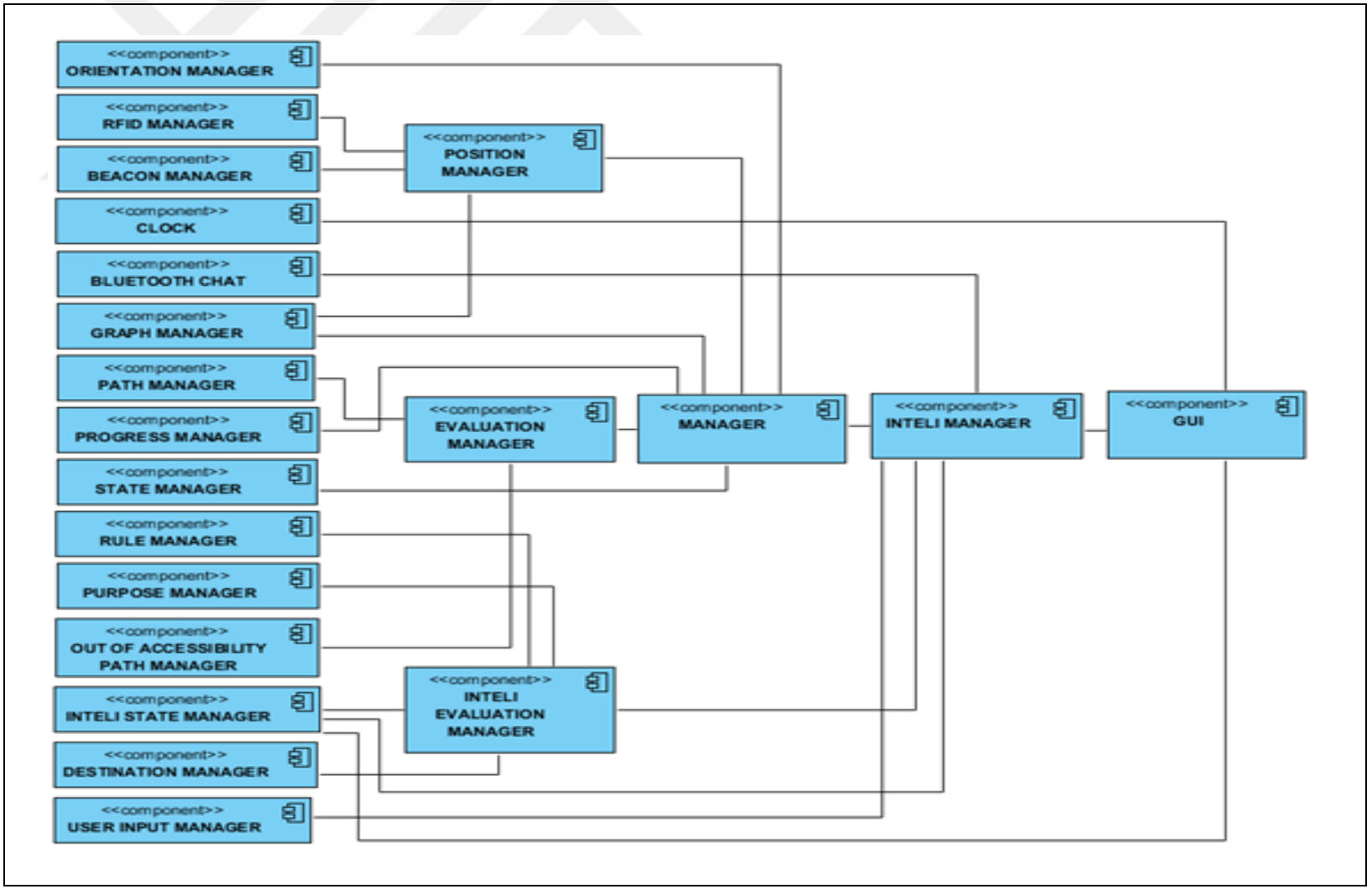


Figure 24: Component view of the system.

The system architecture has similarities with C2 style but there is no strict hierarchy between layered components. A component can send notification to more than one below label component and similarly a below label component can request any service from more than one top level component. Each component is governed by a singleton object. The coordination and access point of the component is the singleton object.

The system includes *Clock, RFID Manager, Rule Manager, Beacon Manager, Graph Manager, Evaluation Manager, Manager, State Manager, Inteli Evaluation Manager, Inteli Manager, Inteli State Manager, Orientation Manager, Path Manager, Progress Manager, User Input Manager, GUI* as main components. Main connectors in the system are *Procedure Call, Event* and *Messaging* type connectors. The implementation of main system components can be seen in Appendix A, Figures 48-52.

4.5 User Interface and Sequence Diagrams

GUI of the system contains one activity (window), divided into four regions as seen in Figure 25. Upper part of the window is for purpose selection and divided into two regions. On the left hand side of the upper part, user can choose the purpose and on the right hand side, user is capable of choosing the purpose condition, if exists. Purpose condition is essential for some purposes. For example, if user has an appointment in the eye department, the user purpose is “medical examination” and the purpose condition is “eye”. On the other hand, purpose condition is not be required if the purpose of the user is payment. After choosing the purpose from the left upper part, the system displays the purpose conditions on the right hand side of the upper part if they exist, and if not, the system leaves this part empty.

The below part of the screen is for commands and is divided into two regions. On the left hand side of the below part of the window, user can push the “Evaluate Route”, “Approve Destination”, and “Repeat the Last Message” buttons which are essential for usage. On the right hand side of the bottom screen, user can choose the “Cancel Route”, “Skip Destination”, and “Pause Navigation” buttons which are needed for special and extraordinary cases as seen in Figure 25.

Talkback application runs parallel with the system. This application reads the screen to the user. User can discover the GUI components by clicking the screen and double click to select any item or activate any button. Alternatively, user can discover the GUI elements by scrolling the screen.

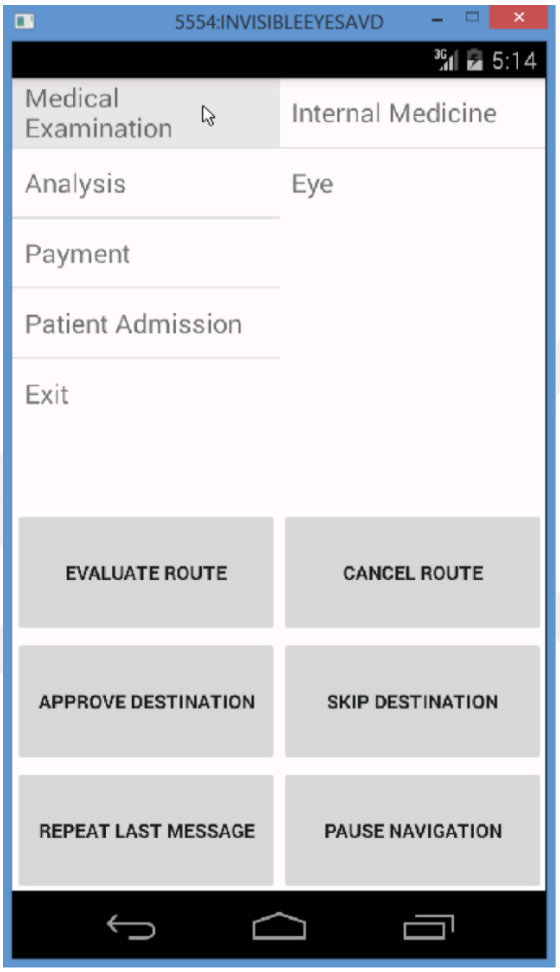


Figure 25: Graphical user interface of the system.

The sequence diagram given in Figure 26 displays the sequence of operations for a purpose containing three destinations considering *GUI*, *Inteli Manager*, and *Manager* components. As seen in Figure 26, sequence starts with specifying a purpose such as “medical examination, internal medicine” by using *GUI* component. *GUI* sends message to *Inteli Manager* and *Inteli Manager* performs the required evaluations. *Inteli Manager* commits message to GUI meaning the evaluation is finished and GUI asks user to accept the first destination point. At the same time, the

user accepts the destination point, and GUI informs the Inteli Manger accordingly. At that time, *Inteli Manager* calls *Manager* to execute obligatory calculations and navigates user to the first destination. When user reaches the first destination, the *Inteli Manager* asks user to approve the next destination by sending a message to *GUI*. The same sequence runs for adjacent destinations and when all the three destinations are completed, user is informed that the route is completed.

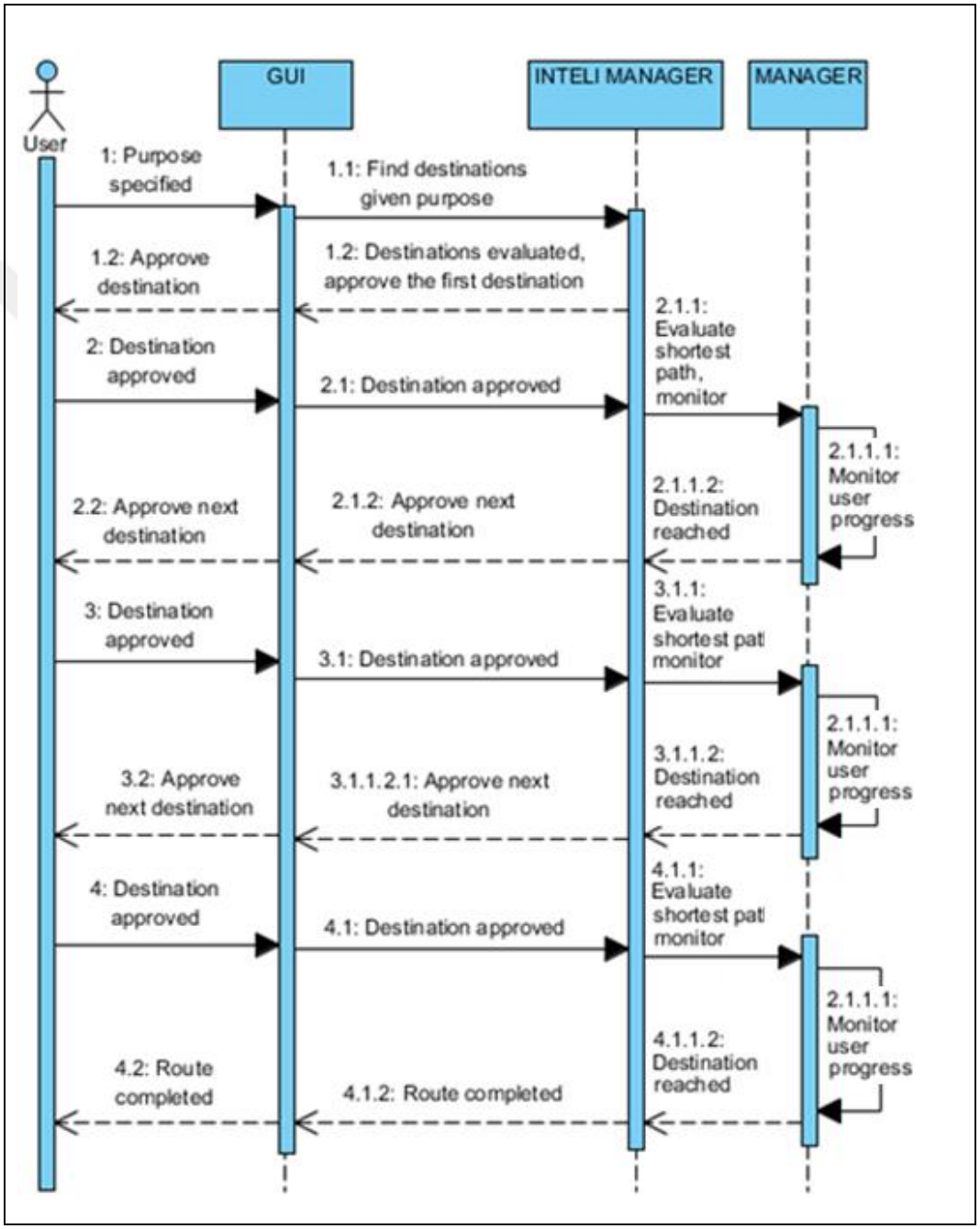


Figure 26: A sequence diagram illustrating sequence for a purpose containing three destinations.

4.6 The System as a Real Time System

The system is designed as a real time system in which the time at which the output is produced is significant. Correctness of the system depends both on correctness of the output and its timeliness. The system runs on discrete time steps, named 200 msec. The implementation of Clock component can be seen in Appendix A, Figure 47 and the component triggers corresponding components at each discrete time step. The state of interaction automaton and navigation automaton is evaluated and the required state transitions are performed at every discrete time step.

4.7 Knowledge Engine and Destination Generation Mechanism

The system has a knowledge engine and destination generation mechanism to determine and order the destination points, given the purpose, as described in the previous chapter. The knowledge engine contains rules about the hospital and is implemented by *IRule* interface, and *Rule* and *Destination* classes. The *Rule* class has a purpose, purpose condition, and destination list. Destinations are expressed by *Destination* class and the *Destination* class has three fields which are *destinationType* representing the type of the destination, *destinationInfo* including information about the destination point, and *destinationNodeID* uniquely maps the destination point to related node.

The inheritance relation between the interface and two classes is the most critical part of the implementation which make recursive inclusion of a rule into another possible. The *Rule* and *Destination* classes both implement the *IRule* interface as seen in Figure 27. Rule class has an array of *IRule* type which means that the list can store both *Rule* and *Destination* type objects. Since *Rule* and *Destination* classes implement the same interface, it is possible to add both *Rule* objects and *Destination* objects in the destination list of a *Rule* object. Therefore, recursive inclusion of a *Rule* object inside the destination list of any other *Rule* object is allowed, which can lead to complicated rule construction. To illustrate the idea, assume that three rules, *R1*, *R2*, and *R3* and a destination point *D1* are given to the system. Then it is possible to include

R1, *R2*, and *D1* into the list of the *R2*. By this way, *R3* will become a recursive rule containing the other two rules and the destination point.

The *Destination* type represents a node on the graph which can be the start or end point of a route. Although the Destination class implements the *IRule* interface, it is not a rule but can be seen as a rule having only one destination point. The destination generation mechanism explained below, will stop recursive execution when all the considered objects are of *Destination* type.

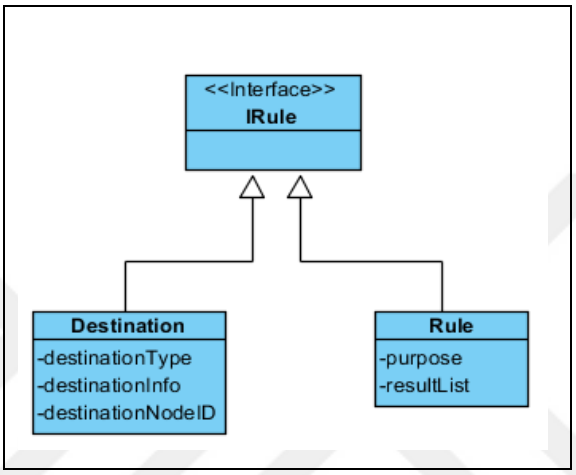


Figure 27: The class diagram representing the super class of *Destination* and *Rule* classes.

For each purpose, a corresponding rule is defined in the system. The mapping between purpose and rule is achieved by encapsulating the corresponding purpose of the rule in *Rule* class as a filed named *purpose*. The *Purpose* class is implemented to represent the purpose of the user. *Purpose* class has *purposeType* and *purposeCondition* fields which are enumerators as seen in Figure 28. The *PurposeTypes* and *PurposeConditions* enumerations define the purposes and conditions that can be inputted by the user.

The design of the algorithms related to destination generation mechanism of the system are given in the previous chapter. The mechanism and related TSP algorithm are implemented. However, it is not possible to find real situation, in our target environment, for a complex rule containing both tight and loose rules together. Therefore, the algorithm for PCTSP, which is required for these cases, is not implemented.

The destination generation mechanism is embedded into the *RuleManager* singleton. The *RuleManager* singleton, initially searches the corresponding rule for a given purpose and current position of the user. After the related rule is found, the *RuleManager* executes the rule by invoking its *Execute* method. The *Execute* method takes a purpose and the current position of the user as input parameters and outputs a list of *Destination* objects which are evaluated and sorted. Execution of a rule means that the rule is converted into destinations and the final list containing only the destinations is sorted as designed in Chapter 3. Afterwards, the *RuleManager* returns the list to the *InteliManager* and the previously described automaton continues.

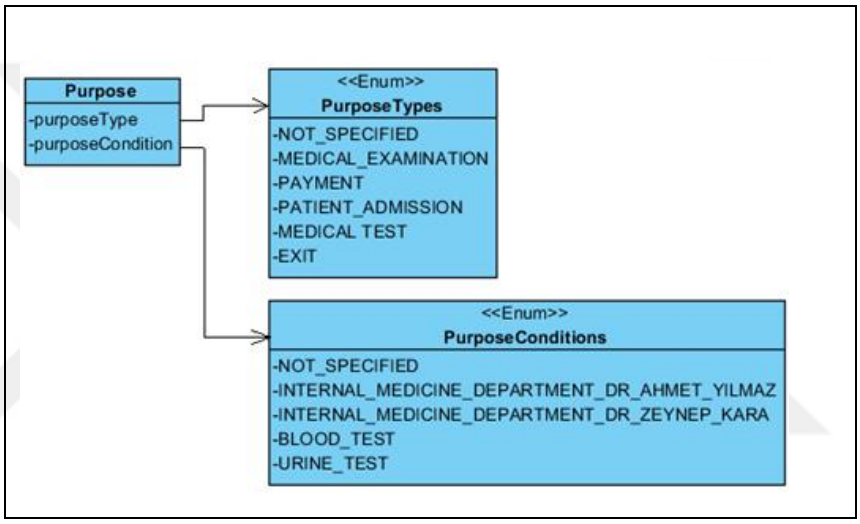


Figure 28: The class diagram of the *Purpose* class.

4.8 Algorithms

The main algorithms are described and designed in the previous chapter. The intelligent part of the system includes the TSP problem and the solution of the TSP is discussed in the previous chapter. Additionally, the *Weighted Multi Lateration Algorithm*, which is the adapted version of the lateration algorithm, is given in detail previously. Apart from the *Weighted Multi Lateration Algorithm* and TSP, the system also includes the shortest path problem which finds the shortest path between two vertices in a graph. The shortest path problem is mentioned but not deeply investigated in the previous chapter because mature solutions are developed for the problem and

these solutions are sufficient for our study. The graph representing the accessibility path is a typical graph which makes possible to implement all the shortest path algorithms. The orientation of the user is also critical for the system. Therefore, orientation algorithm implemented for this purpose is mentioned in this section.

In this section, firstly the shortest path algorithms and the implementation are mentioned. Then the implementation of the TSP problem, which is critical for the intelligence guidance and navigation, is included. Then the proposed *Weighted Multi Lateration Algorithm* and orientation algorithm are explained at the end of the section.

4.8.1 Shortest - Path Algorithm

The Dijkstra's shortest path algorithm is implemented which solves the single-source shortest-paths problem on a weighted, directed graph (Cormen, 2005). Dijkstra's shortest path algorithm is not the only way of finding the shortest path. Although the negative weights are not possible in the study, the Bellman-Ford algorithm is a common alternative to the Dijkstra's algorithm with the advantage of allowing the negative weighted edges (Cormen, 2005). Apart from these algorithms, the search algorithms which are invoked in artificial intelligence applications can be used for the shortest path problem. A simple and A^* related algorithm is proposed by Goldberg et al. (2006) which yields significantly short query times.

The efficient algorithms for the shortest path problem are studied by the author, but the Dijkstra algorithm has a quadratic time complexity in terms of the number of vertices in the graph and since the number of vertices of the graph for the targeted environment is limited, the time complexity of the Dijkstra's algorithm meets the needs.

The *PathManager* singleton is called when the shortest path needs to be evaluated. The *PathManager* singleton instantiates an object of *ShortestPathOperations* class as the relation between two objects as seen in Figure 29 and for simplicity, only the shortest path related members of the *PathManager* singleton are shown in the figure. The execute method of *ShortestPathOperations* class is the main method for evaluating the shortest path. The algorithm repeatedly selects the vertex with minimum shortest path by calling the *getMinimum* procedure, adds the

result to the *settledNodes* list, and relaxes all the edges leaving the node by calling *findMinimalDistance* procedure. The algorithm works exactly same as Dijkstra's algorithm. These algorithms are given in pseudo code at Figure 30 and Figure 31.

After all the shortest paths starting from source node are evaluated by the algorithm, the *getPath* algorithm given in Figure 32, can be called to obtain the shortest path to the given target. This algorithm uses the predecessors data structure to find the path from source to the given target node. The *FindShortestPath* algorithm of the *PathManager* singleton charges the *shortestPathOperations* instance which is of *ShortestPathOperations* type to evaluate and return the shortest path between two nodes as seen in Figure 33.

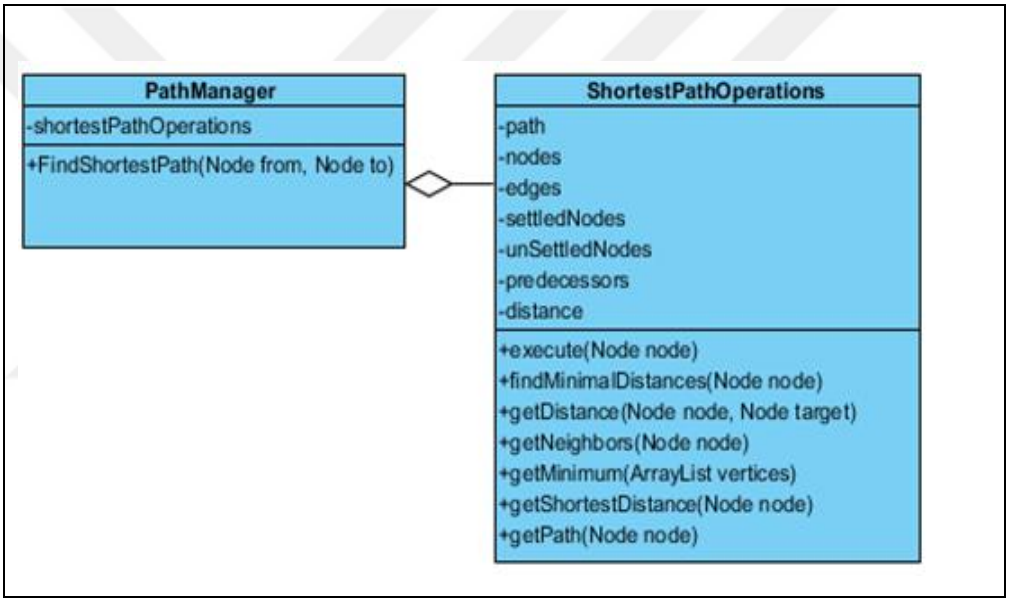


Figure 29: Class diagram representing the relationship between *PathManager* singleton and *ShortestPathOperations* class.

```
Function execute(source)
   distance ← distance ∪ {(source, 0)}
   unSettledNodes ← unSettledNodes ∪{source}
   while length[unSettledNodes] > 0
      do node ← getMinimum(unSettledNodes)
          settledNodes← settledNodes ∪{node}
         unSettledNodes ← unSettledNodes – {node}
         findMinimalDistances(node)
```

Figure 30: The *execution* function of *ShortestPathOperations* class.

```
Function findMinimalDistances(node)
   adjacentNodes = getNeighbors(node)
   for i←1 to length[adjacentNodes]
      do target ← adjacentNodes[i]
         if getShortestDistance(target) >
                        getShortestDistance(node) + getDistance(node, target)
          then distance←distance ∪ {(target, getShortestDistance(node) +
                                          getDistance(node, target))}
               predecessors← predecessors ∪ {(target, node)}
               unSettledNodes ← unSettledNodes ∪ {(target)}
```

Figure 31: The *findMinimalDistances* function of *ShortestPathOperations* class.

```
Function getPath(target)
   step←target
   if predecessor[step] = NULL
      then return NULL
   path←path ∪ {step}
   while predecessor[step] ≠NULL
      do step ← predecessor[step]
         path←path ∪ {step}
   return path
```

Figure 32: The *getPath* function of *ShortestPathOperations* class.

```
Function findShortestPath(fromNode, toNode)
   shortestPathOperations.execute(from)
   shortestPath ← shortestPathOperations.getPath(to)
   return shortestPath
```

Figure 33: The findShortestPath function of *PathManager* singleton.

4.8.2 Traveling Salesman Problem (TSP)

The nearest neighborhood algorithm is used as an approximation algorithm to the Traveling Salesman Problem. The TSP algorithm is implemented in the class named *TSPClass* which is located in the *Algorithms* component of the system. The algorithm is implemented in the *TSP* function of the *TSPClass* class as seen in the Figure 34. The TCP function class NN-TSP function seen in Figure 35 as a sub procedure.

The *TSP* function calls the *GraphManager* and gets the singleton instance. Afterwards, using the instance the graph of the accessibility path of the building is obtained. The distance matrix storing the pairwise shortest path distances between the destinations is constructed by calling the *Shortest-Path-Distance* function. *Shortest-*

Path-Distance function evaluates the shortest path using Djkstra's algorithm and returns the distance or cost of the shortest path given the graph and two node indices. It is also possible to evaluate the pairwise shortest path distances in advance but since every building has its own graph, the system is designed to evaluate these values when required. The NN-TSP algorithm is the place where the nearest neighbor technique was applied for TSP. The algorithm, sorts the nodes sequentially by always finding the nearest node to the last node added to the result list. The Find-Min-Distance-Node function seeks the matrix for a given node and finds and returns the node nearest to the given node.

4.8.3 Precedence Constrained Traveling Salesman Problem (PCTSP)

The PCTSP is not implemented because PCTSP is only required for complex rules which are the combination of tight and loose rules. For our case, these type of complex rules are not required therefore this part is left as a future work.

Genetic algorithms are able to solve the precedence constrained version of the TSP as described in the previous chapter. The genetic algorithm solutions of the PCTSP are studied in the literature and efficient solutions are obtained. Therefore, it will be appropriate to predict the genetic algorithms implementation of the problem in future studies.

```
Function TSP(destinationList, starNode, endNode)
   //Call the GraphManager singleton
   graph_mngr ← GraphManager.getInstance()
   //Get the graph
   Graph graph ← graph_mngr.GetGraph()
   //construct distance matrix
   for i ← 0 to destinationList.Length()
      for j ← 0 to destinationList.Length()
         matrix[i][j] ← Shortest-Path-Distance(graph , i, j)
   //Find the ordered list
   orderedList ← NN-TSP(matrix, startNode)
   if endNode != NULL
      orderedList.Add(endNode)
   return orderedList
```

Figure 34: The *TSP* procedure of the *TSPClass* class.

```
Function NN-TSP(matrix, startNode)
   //Create an empty ordered list
   orderedList ← new ArrayList()
   //Add startNode to the list
   orderedList.Add(startNode)
   //Put the other nodes according to NN
   while orderedList.Length() < matrix.Length()
      do nextNode ← Find-Min-Distance-Node(matrix, startNode)
      orderedList.Add(nextNode)
   return orderedList
```

Figure 35: The NN-*TSP* procedure of the *TSPClass* class.

4.8.4 Positioning algorithms

A modified version of the lateration algorithm named *Weighted Multi Lateration Algorithm* is proposed and explained in previous chapter. Since the algorithm is implemented in the design stage as a proof of the technology and detailed description is given, it is not repeated here. In order to decrease the error rate of the positioning, the average of multi signal strength measurements from nearby Bluetooth beacons are considered.

The position of the user, when he/she is on the accessibility path, can be determined by RFID. Since the position of each node is known, the RFID dependent positioning is implemented as simple SQL queries.

4.8.5 Orientation algorithms

The orientation of the user provides us the data about where the user is facing. This measurement is crucial to navigate user. The basic principle of this prediction is to continuously read the data coming from the magnetic sensor or compass of the fixed mobile device on the waist of the user.

The libraries of Android related to the magnetic field readings are used for this purpose. The class diagram in Figure 36 explains the relation between components related to orientation evaluation mechanism. The *SensorManager* is provided by Android software development kit (SDK) and is the library related with all the sensors of the device. The *MainActivity* implements the *AccelerometerListener* interface

which is also an instance of the *AccelerometerManager* class. The MainActivity and AccelerometerManager are connected to each other and MainActivity listens to the sensors indirectly. The MainActivity is informed when the measurement is changed via *onOrientationCanged* method of *AccelerometerListener* interface. A similar architecture is applied in all the sensor related mechanisms like Beacon and RFID.

The *SensorManager* class in the Android library computes the azimuth, pitch, and roll values. Firstly, it computes the rotation matrix on the basis of gravity and magnetic vector. Once the rotation matrix is calculated, these can be converted to Euler angles representation (Slabaugh, 1999). This part is achieved by SensorManager which is defined in the Android library and we did not change the internal algorithm or mechanism of library. The azimuth value is used to compute the heading of the user. Azimuth is defined as a horizontal angle measured clockwise from the north base line similar to celestial navigation (Rutstrum, 2000). In orientation calculations, the azimuth measurement coming from the device and the north-south direction of the building are compared to determine the orientation of the user.

4.9 Dealing with Sensor Data

During the *Monitoring* state of the *Navigation Automaton*, data coming from sensors is collected and evaluated by the system. Sensor data including Bluetooth beacon data, RFID data, and orientation data as related classes are given in Figure 37. The three data objects are briefly described in the next subsections.

RFID Data

The *RFIDManager* includes a queue to store the RFID data. RFID data includes unique ID and discovery time. Last N_{RFID} values are stored in the queue and if the number of objects in the queue exceed N_{RFID}, the oldest ones are deleted by *RFID Manager*.

IBeacon Data

BeaconManager is responsible for storing the beacon data in a queue for beacon data. Beacon data includes the major and minor id of the beacon, and RSSI value of the

beacon. Last N_{BEACON} values are stored in the queue and if the number of objects in the queue exceed N_{BEACON}, the oldest ones are deleted by the *Beacon Manager*.

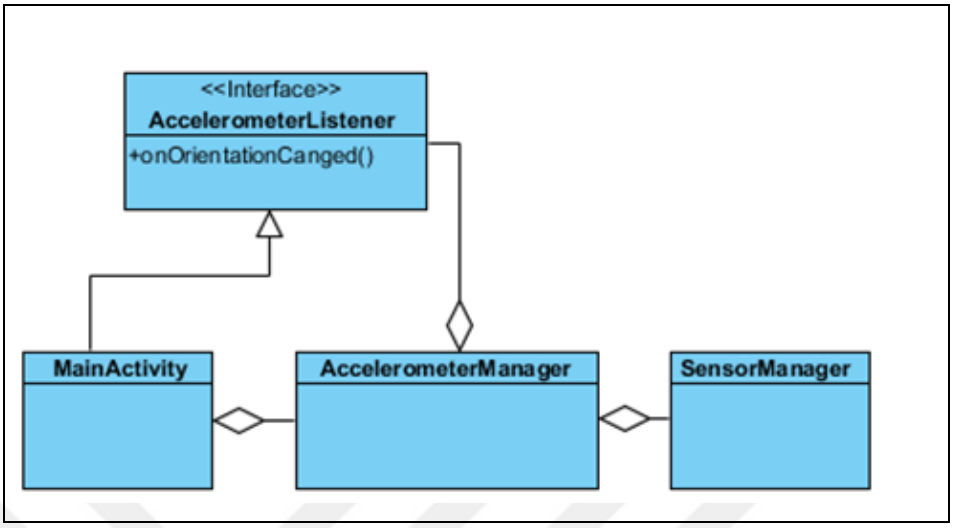


Figure 36: The listener mechanism between *MainActivity* and *AccelerometerManager*.

Orientation Data

OrientationManager captures the instantaneous orientation of the user. Orientation data includes azimuth, pitch and roll of the main mobile device located on the waist of the user. *SensorManager* of the Android is the base component coming from Android SDK used to capture the required data for orientation as described previously in the explanation of the algorithms.

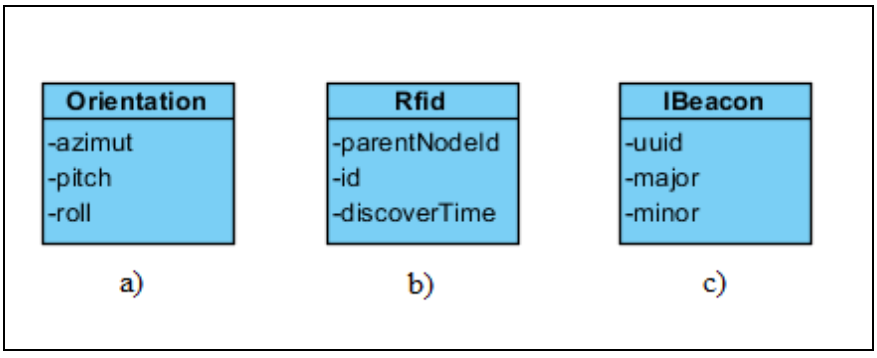


Figure 37: The three classes related to orientation (a), RFID (b), and Bluetooth beacon (c) data.

4.10 Concluding Remarks

The implementation details about Invisible Eyes are given in this chapter. In the implementation stage, the design artifacts produced in the analysis and design stage are considered. Regular meetings with the participatory design team are continued during the implementation stage. The software development reviews are presented to the team and their suggestions and complaints were noted. The software development is achieved in short cycles as agile software development methodologies suggest. At the end of each cycle, a partially working prototype which can be named semi working prototype is generated.

Initially, the general components and the hierarchy between components are implemented as an empty skeleton of the system. In each of the software development cycles, the targeted part of the system is considered and implemented. In the early cycles, the skeleton is converted to a real time system by the inclusion of a clock component and considering the two automaton which explains the main coordination between components. Then, the algorithms including shortest path, TSP, orientation, and positioning are implemented in subsequent software development cycles. Additionally, the sensor related mechanism are implemented and system integration is completed.

In the next chapter, the testing and validation of the implemented system are explained in detail.

Chapter 5

TESTING AND VALIDATION

In this chapter, results of the system test and validation including observations, measurements, and test group contribution will be discussed in detail. In the first part of the chapter, test purposes are defined and methodologies are explained. Then, system features are pointed out and test cases, which are designed to cover test purposes, are introduced. In the next two sections, test environment and test group are described where the test environment is a representative environment for target hospital environment. The last part of the chapter is devoted to tests and test results in which the test results are examined addressing the test purposes and methodologies given in the first part of the chapter. The chapter ends with conclusion summarizing the testing and validation.

5.1 Test Purposes and Methodology

The purpose of the tests performed is to measure the effectiveness of *Invisible Eyes*. Effectiveness of the system is investigated as the capability of meeting the predefined test purposes (TPs). The system features are mainly divided into two groups as external system features (ESFs), and internal system features (ISFs). ESFs are directly related to use of the system and user is aware of them. On the other hand, ISFs are hidden from the user but are crucial for the functionality of the system. The TPs designed for both ESFs and ISFs are defined below:

- *TP1*: Is there any difference for users to navigate in representative environment when the system is present and when users are left alone? What is the probability of reaching the destination points when users are provided with the system and when users are alone without any navigational tool or help?
- *TP2*: What is the success level of ESFs of the system? These ESFs are listed below:

- *ESF1*: The property is related to specifying a purpose by using the GUI. The usability of the GUI in terms of specifying the purpose.
- *ESF2*: The system is controlled by the user by giving six instructions using the corresponding six buttons. The usability of the GUI in term of giving instructions to the system like “Evaluate Route”, “Approve Destination”, “Repeat Last Message”, “Cancel Route”, “Skip Destination”, and “Pause Navigation”.
- *ESF3*: This system property aims to satisfy coordination in working of the two mechanisms which are “the purpose selection mechanism” and “the system controlling mechanism”. The system must be able to inform and guide user when needed to provide coordination between these two mechanisms. This guidance is provided by giving short and informative instructions when needed like “purpose is selected, evaluate the route by using evaluate route button”. The usability of the GUI, considering informing and guidance in coordination between these two mechanisms.
- *ESF4*: The system must give a beep sound when user is on the accessibility path. Success in following the accessibility path by beep sound as positive stimulus when user is on the accessibility path.
- *ESF5*: The mobile device which is on the hand of the user must vibrate when the user is not on the accessibility path and cannot be recognized by the RFID module of the system. Success in following the accessibility path by vibration as negative stimulus when user is not on the accessibility path.
- *ESF6*: The system must re-evaluate the shortest path for the current destination point when user goes to wrong direction and leaves the route (but is still on the accessibility path). Success in re-evaluating the route in such a case.
- ESF7: The system, must guide user back to the accessibility path when user leaves the accessibility path and is not on the path for a while. Success of the system in guiding user back to the accessibility path in such cases.

- *ESF8*: The system must deliver needed information about the destination point before starting to navigate the user to the point. Success of the system in giving required information to users about destination points.
- *ESF9*: The system evaluates and puts in order the destination points given the purpose. User acceptance of the idea is that, users specify the purpose, the system converts it to the corresponding destination points, and navigates users to these destination point following the evaluated order.
- *ESF10*: The system gives short and stepwise navigation instructions which can be easily followed by the user. Success of the system in guiding users by giving alike navigation instructions to a destination point.

- *TP3*: What is the effect of the mobile device using skills of users on their performance in using Invisible Eyes? Users are divided into two groups as advanced mobile device users, and beginner-moderate mobile device users. The average time spent to specify a purpose is measured for each user in both of the groups. Hypothesis testing is performed and statistical significance is evaluated.
- *TP4*: Is there any difference in speed of the users for initial task and for the next task?

TP5: Are both of the critical ISFs satisfied? Apart from tests performed with users which are directly related to ESFs, also artificial cases are generated and two ISFs are tested. These ISFs are listed below:

- *ISF1*: Shortest path generation algorithm and mechanism of the system must be able to provide the shortest path between given two points in all the cases.
- *ISF2*: The system must be able to produce and order the destination point, obeying the destination generation mechanism described previously, given the purpose. This internal system feature is tested for simple rules and the case for composite rules is not tested.

5.1.1 Test Cases

The system test is performed with four test cases which suits with test purposes defined above. All the cases are tested on a predefined representative environment containing representative corridors and rooms. These test cases are given below:

Case1: User has a purpose including a single destination point. System navigates user to this single destination point. User is required to obey the navigation commands produced by the system.

Case2: User inputs a purpose containing multiple destination points. System evaluates and puts in order the destinations using the predefined rules. System sequentially introduces the next destination point to the user and navigates user to this destination point. User is required to obey the navigation commands produced by the system.

Case3: A purpose containing a single destination point, is inputted to the system by the user. In a random point on the path, user is given a pre-defined perturbation and location and direction of the user is changed. The system is required to recalculate the path and navigate the user according to the new path.

Case4: User inputs a purpose including multiple destinations and the system evaluates the destination(s) and navigates user. In a random point, the user leaves the accessibility path and the system is expected to guide the user to reach the nearest point on the accessibility path.

5.1.2 Methodology

A survey including the measurement guide is prepared for testing the system effectiveness. The evaluation survey based on these cases is included in Appendix B. The ESFs defined above are measured with related questions in the questionnaire. The questionnaire is applied on the users after they have tried the system with the four cases. In the questionnaire, a scale of 1 to 5 is used to represent the response of the users. In this scale, the distance between any pair of contiguous numbers is the same. For example, the distance between 1 and 2 is exactly same as from 2 to 3. The tests and test results are analyzed carefully with the help of participants and a general

criterion of accessibility was defined for the evaluated items. Depending on this analysis, 3 is set as acceptable value on a scale of 1 to 5.

Hypothesis testing is applied for the investigation of the validity of the system. The main philosophy of hypothesis testing is to investigate whether a result is obtained by chance or there is a significance of the result. An effect is tested by defining a null hypothesis. The null hypothesis is a model based on the assumption that the apparent effect was actually due to chance (Downey, 2011).

The hypothesis testing in the work is related to find the significance of difference between two mean values and difference of a mean with predefined acceptable threshold value. For *TP2*, the mean values will be compared with predefined acceptable threshold value. One sided test, which asks whether the first mean is significantly higher than the second mean (acceptable threshold value.), is applied. Null hypothesis (H_0) and alternative hypothesis (H_A) for *TP2* can be defined as:

$$H_0: m \leq M \qquad H_A: m > M,$$

where M is the predefined acceptable threshold value. On the other hand, for *TP2* and *TP4* the difference between means of two groups is investigated. Null hypothesis (H_0) and alternative hypothesis (H_A) for *TP3* and *TP4* can be defined as:

$$H_0: \mu_1 = \mu_2 \qquad H_A: \mu_1 \neq \mu_2,$$

where μ_1 is the mean of the first group and μ_2 is the mean of the second group. Two sided test, which asks whether the first mean is significantly different than the second mean, is applied.

The hypothesis that the ESFs are above the acceptable threshold value and the existence of the difference of means of two groups is evaluated by t-test and significance of the results are listed. The t-test gives acceptable power for extremely small sample sizes provided that the effect size is large as stated by Winter (2013). Therefore, t-test is applied to calculate difference between means. The Cohen's d effect size is evaluated and the effect size is found to be great.

Additionally, the observation results and user opinions are listed and the common points are classified and emphasized. These common points are very strong indicators for the evaluation of the system and are good starting points for further studies.

An artificial map (graph) is produced which contains 30 destinations and corresponding 35 rules are defined to test the ISFs. For *ISF1*, the shortest path generation mechanism of the system is tested for different starting and destination points containing multiple possible paths. Similarly, for *ISF2*, different simple rules containing multiple ordered and unordered destination points are executed by the system given the artificial purposes. These trials are repeated many times for both *ISF1* and *ISF2*.

In the next section, the test environment and test users are described followed by performing the tests and test results.

5.2 Representative Environment

Testing and validation are performed with the help of Solvent Software Company. The parking area of the company building is used as the representative environment which is 17 m long and 14 m wide. The scaled indoor map of the parking are can be seen in Figure 38. The parking area is used to represent a hospital environment having seven destination points. Nodes on the accessibility path can be seen in Figure 38 and some pictures of the environment and accessibility path are given in Figure 39.

The destination points are designed to represent the general procedure of the majority of Turkey hospitals. The destination points are numbered as:

1: entrance,

2: patient admission,

3: payment,

4: eye inspection (doctor room),

5: internal medicine inspection (doctor room),

6: blood test center,

7: urine test center.

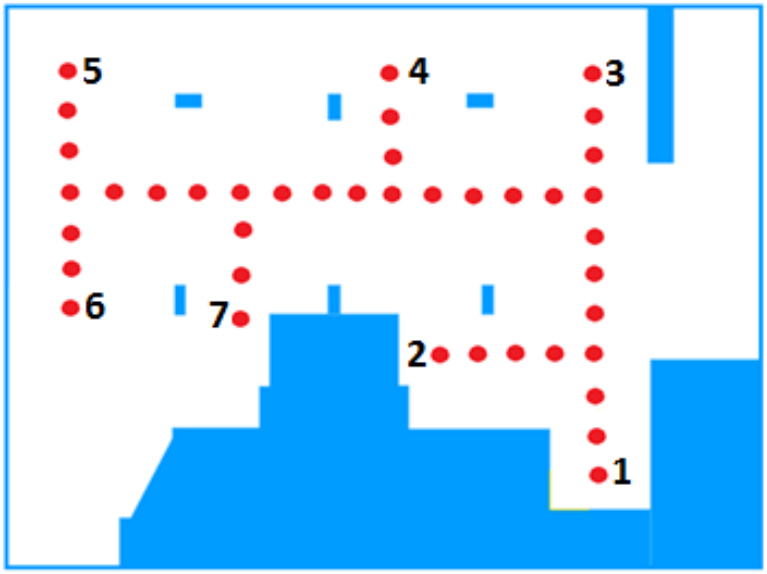


Figure 38: Scaled map of the representative environment and nodes on the accessibility path.

Figure 39: Pictures of representative environment.

5.3 Test Group

We worked with the institution for the blind and visually impaired named "Aktif Görme Engelliler Derneği" to form a test group. The test group contains independently selected 7 individuals whose general characteristics are listed in Table 3.

It is possible to work with blindfolded individuals instead of blinds. In the study of Takatori et al. (2006), 10 blindfolded subjects were tested, and in the work of Seto et al. (2009) they also worked with 3 blindfolded subjects. On the other hand, as we can see in the work of Guerrero .et al. (2012), it is also possible to work with both blinds and blindfolded people. They worked with 2 blinds and 7 blindfolded engineering students.

Table 3: General characteristics of the subjects in test group.

User	**Gender**	**Age**	**Education Level**	**Disability**	**Accessibility tools usage**	**White cane**
User1	male	54	university	100% blind	advanced	advanced
User2	male	35	university	100% blind	advanced	advanced
User3	male	51	less than high school	100% blind	beginner	moderate
User4	male	48	university	100% blind	moderate	moderate
User5	female	43	high school	5% eyesight	advanced	moderate
User6	male	58	less than high school	100% blind	no experience	moderate
User7	male	48	high school	100% blind	beginner	moderate

We preferred to work only with targeted blind users including individuals with different personnel characteristics, so we tried to include as many blind individuals as possible in our test group. We did not prefer to work with blindfolded subjects because even if this type of work will generate quantitative data, blindfolded subjects will not

be able to represent visually impaired people accurately and the results will be unreliable.

5.4 Performing the Tests

We tested the system with our test group described in the previous section. *User1* was also included in our participatory design team and we spent long hours with him during the design and testing. Figure 40 contains some pictures of our tests with *User1*.

We developed four scenarios covering four cases described at the beginning of the chapter "Testing and Validation". We requested the users to perform the scenarios using the system and the progress of the users is observed during these tests. At the end of the test, we asked users to fill a questionnaire given in Appendix B. The snapshot of the tests performed can be seen in Figures 40, 41, and 42.

Figure 40: Snapshots of the test works with User1.

All the test were conducted individually and the questionnaires were applied to the user immediately after finishing the test in an isolated room without any contribution from another participant. The questionnaire is aimed to evaluate the

intelligent purpose selection, destination generation depending on the rule engine, navigation capabilities, and GUI components of the system.

5.5 Test Results

In this section, test purposes (TPs) defined in "Test Purposes and Methodology" section, are attempted sequentially by defined methodologies.

Figure 41: Snapshots of the test works with User2, User3, and User4.

The difference between navigating in the representative environment by using the system and without any system or assistance as defined in *TP1*, gives the immediate result related to the effectiveness of the system. For this measure, we asked

the users whether they are able to accomplish the given tasks without any system or help. They did not want to try the case and emphasized that they cannot navigate to any of the destination points without assistance. On the other hand, all the users were able to navigate to all of the destination points by using the system.

This result immediately implies that use of the system has a strong effect compared to not using any system or help. Since there are no similar systems available that can be applied to the test group, it is not possible to compare the system with another similar system. However, the emphasized system features and effectiveness measure can be used in further similar studies for comparison.

Figure 42: Snapshots of the test works with User5, User6, and User7.

The second test purpose was related to the success level of ESFs. The threshold value for success is defined as 3 out of 5. The questionnaire results can be seen in Table 4 and Table 5. According to the Table 4, 5, it can be concluded that, except *ESF7* al the other ESFs are satisfied and the system is found to be useful for all the asked environments other than "Library" with %95 confidence.

Table 4: Questionnaire results related to the evaluation of the system features.

ESF No	**Mean (1-5)**	**Greater than success level (3)**	**p-value**
1	4,1	yes	0,00233
2	3,9	yes	0,00841
3	3,6	yes	0,00860
4	4,1	yes	0,00233
5	4,3	yes	0,00022
6	3,9	yes	0,00841
7	1,7	no	
8	4,3	yes	0,00205
9	4,1	yes	0,00233
10	3,9	yes	0,02264

Table 5: Questionnaire results related to the possible application areas of the system.

Environment	**Greater than success level (3)**	**p-value**
Hospital	yes	0,01755
University	yes	0,03097
Library	no	
Government buildings	yes	0,00150
Shopping center	yes	0,04194

The reasoning of this evaluation study can be summarized as below:

- Addressing *ESF1*, *ESF2*, and *ESF3*, the GUI was understood by users. Users were able to use the system with the implemented GUI. Users think that, the purpose selection part of the GUI is effective and will be suitable for all the target environments. Apart from purpose selection mechanism, the designed six buttons were successfully used by all the subjects. Additionally, the informative instructions given by the system to provide coordination among purpose selection and system control mechanisms as targeted by *ESF3*, were found to useful by users.

  Related to *ESF4* and *ESF5*, the beep sound (when user is on the accessibility path as positive stimulus) and vibration (when user leaves the accessibility path as negative stimulus) given to inform user to follow the accessibility path, makes it possible to stay in the accessibility path. Users were able to follow the path with these positive and negative stimuli.
- Success in re-evaluating the route when user goes in the wrong direction and leaves the route was the feature aimed at *ESF6*. Users were able to follow the navigation instructions given by the system. The system re-evaluates the route when user leaves the route and goes to a wrong node on the accessibility path. Navigation instructions given in this case were also understood by users and they were able to follow them. This external system feature (*ESF6*) was approved by all the users and found to be successful.
- As described in the previous sections, the external system feature *ESF7* aims to direct users back to the accessibility path when they leave it and the position cannot be recognized by RFID system. In this case, system listens to the data coming from the surrounding Bluetooth beacons and tries to determine the position of the user based on this data. If the approximate position of the user is determined from the data coming from Bluetooth beacons, system gives the required navigation instructions to put the user back to the nearest node on the accessibility path. This property of the system is also tested in our testing phase but the expected success rate was not observed. The error rate of determining the approximate position of the user with data coming from near Bluetooth beacons causes the system to give confusing navigation instructions and

misleads the user. This property of the system will be useful in wide environments and when the user is out of the accessibility path with a great distance. In this case, the errors of Bluetooth beacons will not cause the system to give conflicting navigation instructions.

- According to *ESF8,* the system gives related information before navigating to the next destination point. This external system feature was approved and found to be critical by test group.
- The system allows users to specify the purpose and evaluates the required destinations points as described in the previous sections. This intelligent guidance property of the system, which is the consideration of the *ESF9,* was approved by the users and they liked this feature.
- *ESF10* is related to giving navigation instructions to the user. These short and frequent navigation instructions gave confidence to users and the users were able to follow the route by these instructions.
- According to the subjects, the system can be used in hospitals, universities, government buildings and shopping centers but it may not be successful in libraries. The reason behind this is that the libraries are silent environments and the sound and vibration outputs of the system may not suit the quite surroundings. In tests, all the navigation instructions are given by speakers of the mobile device to be able to observe the subjects. However, headphones can be used and the system can be suitable for silent environments.

The effect of mobile device using skills of users, on the performance of using the system was investigated by *TP3*. Users are divided into two groups as advanced mobile device users, and beginner-moderate mobile device users. The advanced mobile device users group includes *User1*, *User2*, and *User5,* whereas the other users are included in the beginner-moderate mobile device users group.

The average time spent to specify a purpose is measured for each user in both of the groups. There is significant difference between groups, in the average time spent to specify a purpose, with 95% confidence as listed in Table 6. This result implies that, user's mobile device using skils have a significant effect on their performance using Invisible Eyes. Therefore, training the users about mobile device usage can be considered as an initial step.

Table 6: Average time spent to specify a purpose by two groups.

	The average time spent to specify a purpose (sec)
advanced mobile device users	12
beginner-moderate mobile device users	26

Table 7: Speed of users in initial and next task.

	Speed (m/sec)
Initial task	0,2
Next task	0,5

Questioning the difference in speed of the users for initial task and for the next tasks is aimed by *TP4*. There is significant difference between groups, in speed of users in first and next trials, with 95% confidence as listed in Table 7. This results shows that, user speed will increase when they become experienced in using the system.

The test purpose *TP5* investigates the internal system features *ISF1* and *ISF2*. Artificial cases are generated to test *ISF1* and *ISF2*. To investigate the shortest path generation mechanism of the system, 10 different starting and destination points containing multiple possible paths, on an artificial map (graph) containing 30 destinations, are given to the system. In all of the 10 cases, the system evaluated the shortest path among all the alternative paths as expected. Similarly, for *ISF2*, 10 artificial purposes are tested for a total of 35 rules. Purposes are constructed to contain multiple ordered and unordered destination points. The system generated and ordered the destination points considering 35 rules and all the results were the optimum ones that minimizes the total travel distance. These trials were repeated many times for both *ISF1* and *ISF2* and all the results were consistent.

Apart from the questionnaire, observations, and measurements, we also conducted an interview and asked their opinions and suggestions about the system. The results of these interviews are listed below.

User1: We worked with User1 for a long time starting from the design phase as mentioned previously. In the initial test with User1, only a weak beep sound was given

on the RFID reader and the positive and negative stimulus as beep sound and vibration were not present. The user was not able to follow the accessibility path with this weak beep sound. Consequently, we decided to give the beep sound to the speakers and include vibration as negative stimulus. This inclusion made a major positive impact on the success of the system.

Initially, the system was using 12 clockwise directions to navigate users. However, in this case, the system gives very detailed navigation instructions and following these detailed instructions becomes difficult for the user. For example, the navigation instruction "turn left in the 11 clockwise direction" is too detailed and the user is overloaded by such instructions. Therefore, at the end, we decided to give four basic directions instead of 12 clockwise directions.

Additionally, User1 suggested to design a voice user interface in which user will be able to give vocal commands instead of selecting buttons.

User2: This subject suggested to use vibration as a positive stimulus and to use beep sound as negative stimulus. Hearing beep sound continuously annoyed the user.

The user also suggested to embed the system into the white cane and not to use any mobile device. He said that, holding both the mobile device and the white cane is not practical and limits the mobility of the user.

The last suggestion of the user was that the system should be strengthened with an additional system which informs the user about the close surrounding of the user. Information instructions like "toilet is in your left", "there is board in front of you", and "there is waiting area on the right" will increase the understanding of the user about near environment. This is not the target aim of the system but obviously will increase the effectiveness of the system.

User3: He said that using the mobile and white cane at the same time is very hard. Headphones are required according to the user. Additionally, the user liked the property of the system which allows user to specify their purpose instead of directly entering the destination point.

User4: Using the system in noisy environments may not be easy and headphones are required according to the user. The accessibility path must be wider and must be supported with physical stimulus as a raised line. All the hardware must be embedded on the white cane. Information about all the destinations can be given before the users

starts the path. The system gives navigation instructions on each node and if the user is following a straight path the navigation instruction “go straight” is not necessary. Navigation instructions at the beginning of straight paths and at turn points will be sufficient.

User5: The user emphasized that, these systems are required also for outdoors. Additionally, similar systems are required for mass transport systems and must be widely available around the country.

User6: He suggested to embed the system on the white cane and emphasized the importance of these systems.

User7: He said that the cables must be avoided and the system must be embedded to the white cane. Additionally, if mobile devices are included to the system these devices must be small in size. The system gives the navigation instructions when user reaches the turning point but it will be more beneficial if it previously informs the user about the turning points.

The first common suggestion about the system was that, embedding the system into the upper part of the white cane will increase the effectiveness of the system by avoiding mobile device on the hand of the user. By this way, users’ mobility will increase. Similarly, headphones must be included instead of using the speakers of the mobile devices according to the most of the users. On the other hand, intelligent guidance and navigation property of the system was approved by all the users.

5.6 Concluding Remarks about Tests

The system is tested and validation of the system is given as a result of the tests performed. Test purposes are defined addressing the effectiveness of the system and the system features. The system features are divided into two groups as external system features (ESFs) and internal system features (ISFs). External system features are related to the recognizable features that are directly visible to the user. The most critical internal system features are investigated which are hidden from users but still are very important for the system.

The system is tested on a representative environment containing seven destination points. The test group includes seven independently selected blind

subjects. In the test, only blind subjects are included to increase the significance of the work.

Predefined test cases are performed by the users. For each user, progress is observed and performance is measured. At the end of each test, the user is asked to fill a questionnaire in an isolated room where the questions are read to the user and responses are noted.

At the end of the tests, the data is analyzed and results are listed. Majority of the items in the test purposes are found to successful with 95% confidence. The intelligent guidance and navigation mechanism of the system, which was the main objectives of the study, was found to be successful. On the other hand, ESF7, which is related to guiding user back to the accessibility path when user left the accessibility path and is not on the path for a while, failed to satisfy the required success. The reason behind the failure of ESF7 was the error rate of the Bluetooth beacons. The error rate in positioning the user by using data coming from near Bluetooth beacons causes the system to give confusing navigation instructions and misleads the user. Additionally, the system was found to be unsuitable for libraries according to the users. The reason behind this belief was that for test purposes the navigation instructions are directly given via speakers of the mobile device to observe the progress of users and it causes a noise not suitable for silent environments. However, this problem can be solved easily by using headphones instead of speakers.

Apart from observations, measurements, and questionnaires, suggestions and ideas of the users were also collected on individually performed interviews. The common suggestion from these interviews was that, the user interface of the system can be embedded into the white cane and the user can use the system with buttons provided on the white cane. This way, users will be able to use the system without a mobile device on their free hand and mobility of the user will increase as a result.

In conclusion, all the users approved the purpose selection mechanism and intelligent guidance and navigation principle of the system. In future work, new prototypes depending on user suggestions can be implemented and tested.

Chapter 6

CONCLUSION AND FUTURE WORK

The general purpose of Invisible Eyes is to provide visually impaired people with an intelligent guidance and navigation system to improve the quality of life by increasing the mobility and by providing accessibility in indoor environments. The knowledge obtained from a comprehensive literature review provided us with a range of questions and ideas. The main conclusion from the review was the finding that an intelligent guidance and navigation system is crucial for visually impaired people especially for complex indoor environments. Therefore, a purpose specification mechanism is designed, implemented, and tested. Along with the intelligent system the working methodology is also noted as a need for the studies including blind persons. Accordingly, before starting the design, we concentrated on the definition of the methodology and establishing the test environment and all the other required conditions.

The participatory design team including the target users, specialists, software engineers, and project managers is constructed because we understand that this is a basic need and is not emphasized enough in the literature. The participants have attended the regular weekly meetings to review and discuss ideas or progress.

The stages of the work are planned to correspond to the steps defined in generic software development life cycle which are requirement specification, design, build, and test. The agile software construction methodologies are applied meaning the steps are revisited repeatedly according to the needs and progress.

After the requirement specifications are defined with the participatory design team, all the components of the system with techniques and algorithms are designed. The system mainly contains two sub systems which are the main system attached to the waist of the user and the user interface system which is deployed to the mobile device held by the user. The software architecture followed was the C2 style in which components are hierarchically distributed among the levels. Components are designed as singletons because only one instantiation of classes which manage the execution, is

required. Additionally, Façade design pattern is implemented to deal with the internal complexity of the system.

The mechanism which allows the user to input the purpose and enable the system to convert the given purpose to the destination points is designed. The problem related with this mechanism is similar to traveling salesman problem (TSP) and nearest neighbor algorithm is found to be suitable for our case after considering all the algorithms applied for the TSP. The mechanism is designed to be able to deal with complicated, recursive rules. In this case the related algorithm becomes similar to precedence constrained travelling salesman problem (PCTSP) instead of (TSP). The PCTSP is proposed, but not implemented because complex and recursive rules are not valid for the targeted environment and it will not be found an application for our case.

The trilateration algorithm is adopted for imperfect cases in which distance estimations are evaluated in error and the circles are not intersecting. The adopted lateration algorithm considers the distance values as weight and evaluates the weighted centroid as the position estimate.

The prototype of the proposed system is implemented and deployed. The system is tested in a representative environment using the test group which includes seven blind individuals. Several test cases are designed addressing the external features of the system. The results were observed and users were asked to evaluate the system via survey. Targeted purpose selection destination evaluation mechanism is found to be significantly effective at the end of the evaluation. However, the proposed Bluetooth positioning system designed to guide the user back to the accessibility path when the user is out of the path was not effective because of the error rate in measurement. The system tolerates the error rate when user is considerably out of the accessibility path greater than 2 m. but otherwise gives conflicting instructions because of the error rate of Bluetooth measurements. Additionally, the system is found to be applicable for governmental buildings, universities, shopping centers but not for libraries because of the noisy working principle of the prototype in which sound signals are given to the speakers of mobile devices. However, the sound signals could be given to headphones instead of speakers and this drawback can be resolved easily.

Future work needs to be conducted on the system of Invisible Eyes, in order to achieve the optimum system and functionality to be used in real life with guidance

of the highlighted items obtained after the testing and validation stage. The followings are suggestions of the future work:

Embedded Integrated System

Visually impaired subjects contributed to the testing and validation stage suggested to integrate the user interface components to the white cane. In our design, the user interface related components were deployed to a mobile device which is held by the user. However, after the testing and validation stage it is observed that the users' mobility is limited because it was not easy to hold both the device and white cane. Therefore, a special white cane which includes the user interface where the users are able to interact with the system using special buttons and vibrating regions on the white cane is stated as a future work.

Supportive Informative System

The system gives information about the evaluated and ordered destination points before user starts to navigate to the point. The purpose of this information is to get the user familiar with the target destination and understand why he/she is going to this point. In addition to the current system, users suggested to include additional modules to the system which will give information about the surrounding environment of the user. This informative module should give information about the environment when the user requires. Therefore, the user should be able to use the system even without inputting a purpose. With the help of this supportive informative system, users will be able to hear the content of any nearby board where announcements or any other information is provided. The alignment of corridors, rooms, and doors should be explained to the user by converting stored text data to speech, on demand. The supportive system will increase the effectiveness of the system and also will contribute to its commercial value.

Adaptation for Outdoor and Mass Transport

In the dissertation, the environment is defined to be the complex environments like hospitals where defining the purpose is more practical instead of directly choosing the destination points, and is limited to be indoor. However, the outdoor can also be considered and system can guide and navigate users indoor and/or outdoor. This expansion of the system could be achieved easily because of the success of the outdoor position estimation technologies and systems like global positioning system (GPS).

Similarly, the system should be capable of acquiring data related to mass transportation systems from city information system central database. Therefore, users can gain even the dynamic information containing the delays or any updates related to the transportation vehicle like buses and trains.

In conclusion, participatory design and agile methodologies are applied together to analyze, design, implement, and validate Invisible Eyes which aimed to provide an indoor intelligent guidance and navigation system to the visually impaired people for the complex environments like hospitals. Understanding the objective of the user, ensuring the shortest route, and providing stepwise real time navigation instructions are the main considerations of the system and makes it unique in the literature. This is the innovative system which encourages the impaired people to be involved actively into the society, increases their mobility and the quality of the life by providing more accessible environments.

REFERENCES

Alghamdi S., Schyndel Ron van, Khalil I., “Accurate positioning using long range active RFID technology to assist visually impaired people”, Journal of Network and Computer Applications, 2014.

Alsalibi B. A., Jelodar M. B., and Venkat I., “A Comparative Study between the Nearest Neighbor and Genetic Algorithms: A revisit to the Traveling Salesman Problem”, International Journal of Computer Science and Electronics Engineering (IJCSEE) Volume 1, Issue 1 ISSN 2320–4028 (2013).

Altini M., Brunelli D., Farella E., and Benini L., “Bluetooth Indoor Localization with Multiple Neural Networks”,5th International Symposium on Wireless Pervasive Computing (ISWPC), 2010.

Anastasi G., Bandelloni R., Conti M., Delmastro F., Gregori E., and Mainetto G.,“Experimenting an indoor bluetooth-based positioning service” Proceedings of 23rd International Conference on Distributed Computing Systems Workshops, 2003.

Applegate D., Bixby R., Chvatal V., and Cook W., "Implementing the Dantzig Fulkerson-Johnson Algorithm for large traveling salesman problems," Mathematical programming vol. 97, pp. 91-153, 2003.

Au A. W. S., Chen Feng, Valaee S., Reyes S., Sorour S., Markowitz S. N., Gold D., Gordon K. and Eizenman M.; “Indoor Tracking and Navigation Using Received Signal Strength and Compressive Sensing on a Mobile Device”; IEEE Transactions on Mobile Computin, Vol. 12, No. 10, October 2013.

Berenbach B., Paulish D. J., Kazmeier J., Rudorfer A., “Software & Systems Requirements Engineering: In Practice”, Mc Graw Hill, 2009.

Bourque P., Fairley R.E., "Guide to the Software Engineering Body of Knowledge (SWEBOK)". IEEE Computer Society, 2014.

Brunato M., and Battiti R., “Statistical learning theory for location fingerprinting in wireless LANs”, Computer Networks, 2005.

Chawathe S. S., “Beacon placement for indoor localization using bluetooth”, Intelligent Transportation Systems, ITSC 2008, 11th International IEEE Conference, 2008.

Chrysikos T., Gergopoulas G., and Kotsopoulas S., “Site Specific validation of ITU Indoor Path Loss Model at 2.4 GHz”, 10th IEEE International Symposium on a World of Wireless, Mobile and multimedia Networks (WoWMoM), 2009.

Chumkamon S., Tuvaphanthaphiphat P., Keeratiwintakorn P.; “A Blind Navigation System Using RFID for Indoor Environments”; Electrical Engineering/Electronics,

Computer, Telecominication and Information Technology, ECTI-CON 2008. 5th International Conference, 2008.

Cook et al., Cook, B., Buckberry, G., Scowcroft, I., Mitchell, J.E., and Allen, T., 2005, "Indoor Location using Trilateration Characteristics," in London Communications Symposium (LCS05), 1-5, 2005.

Cormen T. H., Leiserson C. E., Rivest R. L., and Stein C., "Introduction to Algorithms", Second Edition, The MIT Press, London, England, 2005.

Dantzig G., Fulkerson R., and Johnson S. "Solution of a large-scale traveling salesman problem," Operations Research Letters, vol. 2, pp. 393-410, 1954.

Diaz J. J. M., Maues R. A., Soares R. B., Nakamura E. F., and Figueiredo J. M. S., "Bluepass: An indoor bluetooth-based localization system for mobile applications" In Computers and Communications (ISCC), 2010 IEEE Symposium, 2010.

Downey Allen B., "Think Stats Probability and Statistics for Programmers", Green Tea Press, Needham, Massachusetts, 2011.

Elbes M., Al-Fuqaha A., "Design of a Social Collaboration and Precise Localization Services for the Blind and Visually Impaired", Procedia Computer Science, 2013.

Feldmann S., Kyamakya K., Zapater A., and Lue Z., "An indoor bluetooth-based positioning system: Concept, implementation and experimental evaluation", In International Conference on Wireless Networks, 2003.

Ferrario, Maria Angela and Simm, William and Newman, Peter and Forshaw, Stephen and Whittle, Jon (2014) Software engineering for 'social good':integrating action research, participatory design, and agile development. In: ICSE Companion 2014 Companion Proceedings of the 36th International Conference on Software Engineering. ACM, New York, pp. 520-523. ISBN 9781450327688.

Filipe V. M., Fernandes F., Fernandes H., Sousa A., Paredes H., Barrosa J., "Blind navigation support system based on Microsoft Kinect" ,Proceedings of the 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2012).

Freeman, Eric; Freeman, Elisabeth; Kathy, Sierra; Bert, Bates (2004). Hendrickson, Mike; Loukides, Mike, eds. "Head First Design Patterns" (paperback) 1. O'Reilly: 243, 252, 258, 260.ISBN 978-0-596-00712-6. Retrieved 2012-07-02.

Gambardella L.M., Montemanni R., Weyland D., "An Enhanced Ant Colony System for the Sequential Ordering Problem", In Operations Research Proceedings 2011, Operations Research Proceedings, pages 355–360. Springer Berlin Heidelberg, 2012.

Gast M. S., “Building Applications with iBeacon, Proximity and Location Services with Bluetooth Low Energy”, O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA, 2015.

Goldberg A.V., Kaplan H., and Werneck R.F., “Reach for A*: Efficient Point-to-Point Shortest Path Algorithms,” Proc. SIAM Workshop Algorithms Eng. and Experimentation, page no. 129-143, 2006.

Gorton I., “Essential Software Architecture”, Springer Berlin Heidelberg New York, 2006.

Guerrero L. A., Vasquez F. and Ochoa S. F.; “An Indoor Navigation System for the Visually Impaired”; Sensors 2012, 12, 8236-8258; doi:10.3390/s120608236; 2012.

Held M., and Karp H. M.,“A dynamic programming approach to sequencing problems”, Michael Held and Richard M. Karp, Journal for the Society for Industrial and Applied Mathematics 1:10. 1962.

Hub A., Diepstraten J., Ertl T.; “Design and Development of an Indoor Navigation and Object Identification System for the Blind.”; In Proceedings of the ACM SIGACCESS Conference on Computers and Accessibility, Atlanta, GA, USA, 18–20 October 2004; pp. 147–152.; 2004.

Kammoun S., Parseihian G., Gutierrez O., Brilhoult A., Serpa A., Raynal M., Oriola B., Mace M.J.-M., Auvray M., Denis M., Thorpe S.J., Truillet P., Katz B.F.G., Jouffrais C.; “Navigation and space perception assistance for the visually impaired: The NAVIG project”, IRBM 33 (2012) 182-189.

Kautz K.; “Participatory Design Activities and Agile Software Development.” Jan PriesHeje; John Venable; Deborah Bunker; Nancy L. Russo; Janice I. DeGross. Human Benefit through the Diffusion of Information Systems Design Science Research, 318, Springer, pp.303- 316, 2010, IFIP Advances in Information and Communication Technology, 978-3-642-12112-8.

Koch J., Wettach J., Bloch E., Berns K.; “Indoor Localisation of Humans, Objects, and mobile Robots with RFID Infrastructure”; Seventh International Conference on Hybrid Intelligent Systems; 2007.

Kohoutek T. K., Mautz R., and Donaubauer A., “Real Time Indoor Positioning Using Range Imaging Sensors”, Proceedings of SPIE Photonics Europe, Real Time Image and Video Processing, vol. 7724., 2010.

Kotonya G., and Somerville I., “Requirements Engineering: Processes and Techniques”, John Wiley and Sons, 1998.

Kusiak A., Finke G., “Modeling and solving the flexible forging module scheduling problem”, Engineering Optimization 12 (1) (1987) 1–12.

Lawler E. L., Lenstra J. K., Rinnooy A. H. G., and Shmoys D. B., "The traveling salesman problem", John Wiley & Sons, 1985.

Lees-Miller J., Box S. and Wilson R. E. (2013) Hidden Markov Models for Vehicle Tracking with Bluetooth. Transportation Research Board, Washington DC, USA, 13-3032 TRB Highway Traffic Monitoring Committee (ABJ35).

Lorenz B., Ohlbach H. J. and Stoffel E. P.; "A Hybrid Spatial Model for Representing Indoor Environments"; 6th International Symphosium (Web and Wireless Geographical Information Systems), 2006.

Marco Altini, Davide Brunelli, Elisabetta Farella, and Luca Benini. Bluetoothindoor localization with multiple neural networks. InWireless Pervasive Com-puting (ISWPC), 2010 5th IEEE International Symposium on, pages 295 –300,may 2010.

Martin C. R., and Martin M., "Agile Principles, Patterns, and Practices in C#", Prentice Hall, Print ISBN-10: 0-13-185725-8 Print ISBN-13: 978-0-13-185725-4, 2006.

Moreno M., Shahrabadi S., Jose J., Buf J. M. H., Rodriges J. M. F., "Realtime local navigation for the blind: detection of lateral doors and sound interface",Proceedings of the 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2012).

Moulton B., Pradhan G., Chaczko Z.; "Voice Operated Guidance Systems for Vision Impaired People: Investigating a User-Centered Open Source Model"; International Journal of Digital Content Technology and its Applications Volume 3, Number 4, December 2009.

Mautz R., and Tilch S., "Optical Indoor Positioning Systems", Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011, Portugal.

Mingozzi A., Bianco L., and Ricciardelli A., "Dynamic programming strategies for the TSP with time windows and precedence constraints", Operations Research 45 (1997) 365–377.

Moon C., Kim J., Choi G., Seo Y., "An efficient genetic algorithm for the traveling salesman problem with precedence constraints", European Journal of Operational Research 140 (2002) 606–617.

Muffert M., Siegemund J., and Forstner W., "The Estimation of Spatial Positions by Using an Omnidirectional Camera System", Proceedings of the 2nd International Conference on Machine Control and Guidance, pp. 95-104., 2010.

Mulloni A., Wgner D., Schmalstieg D., and Barakonyi I., "Indoor Positioning and Navigation with Camera Phones", Pervasive Computing, IEEE, vol. 8, pp. 22-31, 2009.

Na J.; “The Blind Interactive Guide System Using RFID-Based Indoor Positioning System.”; In Proceedings of the 10th International Conference on Computers Helping People with Special Needs, Linz, Austria, 11–13 July 2006; LNCS 4061, pp. 1298–1305.; 2006.

Nikitin P. V. and Rao K. V. S.; “Performance Limitations of Passive UHF RFID Systems”, American Physical Society (APS), Albuquerque, NM, July 2006.

Norman DA., “The Invisible Computer”, Cambridge, Mass: MIT Press, 1998.

Öktem R., Aydın E., Cagiltay N. E.; “An RFID Based Location Finding and Tracking with Guidance”, 2008 IEEE.

Öktem R., Aydın E., “An RFID based indoor tracking method for navigating visually impaired people”,Turk J Elec Eng & Comp Sci, Vol.18, No.2, 2010.

Park J., Charrow B., Curtis D., Battat J., Minkov E., Hicks J., Teller S., and Ledlie J., “Growing and Organic Indoor Location System”, Proceedings of International Conference on Mobile Systems, Applications and Services, 2010.

Park S., Choi In-Mook, Kim Sang-Soo, Kim Sung-Mok, “A portable mid-range localization system using infrared LEDs for visually impaired people”, Infrared Physics & Technology, 2014.

Prattico F., Cera C., Petroni F., “A new hybrid infrared-ultrasonic electronic travel aids for blind people”, Sensors and Actuators A: Physical, 2013.

Ralph P., and Wand Y., “A Proposal for a Formal Definition of the Design Concept”, Design Requirements Workshop, 2009.

Renaud J., Boctor F.F., Ouenniche J., “A heuristic for the pickup and delivery traveling salesman problem”, Computers and Operations Research 27 (2000) 905–916.

Rondriguez-Sanches M.C., Moreno-Alvarez M.A., Martin E., Borromeo S., Hernandez-Tamames J.A., “Accessible smartphones for blind users: A case study for a wayfinding”, Expert Systems with Applications, 2014.

Rutstrum C., “The Wilderness Route Finder: The Classic Guide to Finding Your Way in the Wild”, University of Minnesota Press, ISBN 0-8166-3661-3, 2000.

Ruzyyev A., Hasbullah H., and Subhan F., “A survey of child tracking techniques in wireless sensor networks”, Int. J. Res. Rev. Comput. Sci., 2010.

Ruzyyev A., Hasbullah H., and Subhan F., “Indoor Child Tracking in Wireless Sensor Network using Fuzzy Logic Techniques”, Research Journal of Information Technology, 2011.

Sato T., Nakamura S., Terabayashi K., Sugimoto M., and Hashizume H., “Design and Implementation of a Robusta and Real Time Ultrasonic Motion Capture System”, Proceedings of the 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011, Portugal.

Savelsbergh M.W.P., and Sol M., “The general pickup and delivery problem”, Transportation Science 29 (1995) 17–29.

Serrao M., Rodrigues J.M.F., Rodrigues J.I., Buf J.M.H., “Indoor localization and navigation for blind persons using visual landmarks and a GIS”, Proceedings of the 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2012).

Seto T. and Magatani K., “A navigation system for the visually impaired using colored navigation lines and RFID tags”; 31st Annual International Conference of the IEEE EMBS Minneapolis, Minnesota, USA, September 2-6, 2009.

Sjöberg C. Voices in design: argumentation in participatory development. Linköping Studies in Science and Technology, Thesis No. 436. Linköping, Sweden: Linköping University; 1994.

Slabaugh G. G., “Computing Euler angles from a rotation matrix”, 1999.

Sonnenblick Y.; “An Indoor Navigation System for Blind Individuals”; In Proceedings of the 13th Annual Conference on Technology and Persons with Disabilities, Los Angeles, CA, USA, 17–21 March 1998; pp. 215–22; 1998.

Sung J. and Jeong B., “An Adaptive Evolutionary Algorithm for Traveling Salesman Problem with Precedence Constraints”, Hindawi Publishing Corporation The Scientific World Journal Volume 2014, Article ID 313767, 11 pages.

Takatori N., Nojima K., Matsumoto M., Yanashima K., and Magatani K.; “Development of voice navigation system for the visually impaired by using IC tags”; Proceedings of the 28th IEEE EMBS Annual International Conference, New York City, USA, Aug 30-Sept 3, 2006.

Takenga C. M., Anne K. R., Kyamakya K., and Chedjou J. C., “Comparison of Gradient descent method, Kalman Filtering and decoupled Kalman in training Neural Networks used for fingerprint-based positioning”, Proc. IEEE 60th Veh. Technol. Conf., vol. 6, pp. 4146-4150, 2004

Taylor R. N., Medvidoviç N., Anderson K.N., Whitehead E.J., Robbins J.E., Nies K.A., Oreizy P., Dubrow D., “A Component and Message Based Architectural Style for GUI Software”, IEEE Transaction on Software Engineering, 1996.

Taylor R. N., Medvidoviç N., Dashofy E. M., “Software Architecture Foundations, Theory, and Practice”, John Wiley & Sons, Inc., 2010.

T.C. Aile ve Sosyal Politikalar Bakanlığı, "Erişilebilirlik İzleme ve Denetleme Yönetmeliği ", access date: 10.05.2015 on http://www.eyh.gov.tr/mevzuat/ulusal-mevzuat/yonetmelikler/erisilebilirlik-izleme-ve-denetleme-yonetmeligi.

T.C. Sağlık Bakanlığı, "Sağlık Kurumlarında Özürlü Bireyler için Ulaşılabilirlik Temel Bilgiler Rehberi", access date: 10.05.2015 on http://www.tkhk.gov.tr/Dosyalar/8b9be1a57d364452b8f2f9b03b1de8de.pdf.

Treuillet S. and Royer E.; "Outdoor/indoor vision-based localization for blind pedestrian navigation assistance"; Int. J. Image Graph. 2010, 10, 481–496.; 2010.

Troels Laursen, Nikolaj Bisgaard Pedersen, Jimmy Jessen Nielsen, Tatiana K. Madsen: Hidden Markov Model based mobility learning fo improving indoor tracking of mobile users. WPNC 2012: 100-104

Trucco E. and Plakas K., "Video Tracking: A Concise Survey", IEEE Journal of Oceanic Engineering, vol. 31, no. 2, pp.520-529, 2006.

Tsirmpas C., Rompos A., Fokou O., Koutsouris D., "An indoor navigation system for visually impaired and elderly people based on Radio Frequency Identification (RFID)", Information Science, 2015.

Yun S., Lee J., Chung W., Kim E., and Kim S., "A soft computing approach to localization in wireless sensor networks",Expert Systems with Applications, 2009.

Wan E., and Paul A., "A Tag free Solution to Unobtrusive Indoor Tracking Using Wall Mounted Ultrasonic Transducers",Proceedings of the 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2010, Switzerland.

Ward A., Jones A., and Hopper A., "A new location technique for the active office", Personel Communications, IEEE, vol. 4, no. 5, pp. 42-47, 1997.

Whitehead E. J., Robbinson J. E., Medvidovic N., and Taylor R.N., "Software Architecture: Foundation of a Software Component marketplace", Proc. Int'l Workshop on Architectures for Software Systems, Apr. 1995.

Winter J. S. F., "Using the Student's t-test with extremely small sample sizes", Practical Assessment, Research & Evaluation, ISSN 1531-7714, Volume 18, Number 10, August 2013.

Wong F., Nagarajan R., Yaacob S..; "Application of Stereovision in a Navigation Aid for Blind People"; ICICS-PCM2003; 2003.

APPENDICES

Appendix A: Implementation of Main System Components.

```
public class IBeacon
{
  public String _ID;
  private int txPower;
  private int rssi;
  private int major;
  private int minor;
  private String uuid;
  public double dist;

  public IBeacon fromscanData(byte[] scanData, int rssi)
  {
    IBeacon iBeacon = new IBeacon();

    iBeacon.major = (scanData[25] & 0xff) * 0x100 + (scanData[26] & 0xff);
    iBeacon.minor = (scanData[27] & 0xff) * 0x100 + (scanData[28] & 0xff);
    iBeacon.txPower = (int)scanData[29];
    iBeacon.rssi = rssi;
    byte[] proximityUuidBytes = new byte[16];
    System.arraycopy(scanData, 9, proximityUuidBytes, 0, 16);
    String hexString = bytesToHex(proximityUuidBytes);
    StringBuilder sb = new StringBuilder();
      sb.append(hexString.substring(0,8));
      sb.append("-");
      sb.append(hexString.substring(8,12));
      sb.append("-");
      sb.append(hexString.substring(12,16));
      sb.append("-");
      sb.append(hexString.substring(16,20));
      sb.append("-");
      sb.append(hexString.substring(20,32));
    iBeacon.uuid = sb.toString();
    return iBeacon;
  }

  public void setRssi(int rssi)
  {
    this.rssi = rssi;
  }

  public int getRssi()
  {
    return this.rssi;
  }
  . . .
}
```

Figure 43: IBeacon data object representing Bluetooth beacon data.

```
public interface BeaconListener
{
   public void beaconFound(IBeacon device);
}
```

Figure 44: *BeaconListener* interface for data transfer from *BeaconScanner* to *MainActivity*.

```
import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothManager;
import android.content.Context;
. . .
public class BeaconScanner
{
   private BluetoothAdapter mBluetoothAdapter;
   private static BeaconListener listener;
   private static final long SCAN_PERIOD = 10000;
   private static Context mContext;
   . . .
   public BeaconScanner(Context c, BeaconListener blistener)
   {
      mContext = c;
      mHandler = new Handler();
      listener = blistener;
   }
   public void enableBLE()
   {
      . . .
   }
   public void scanLeDevice(final boolean enable)
   {
      . . .
   }
   private BluetoothAdapter.LeScanCallback mLeScanCallback =
         new BluetoothAdapter.LeScanCallback()
   {
            @Override
            public void onLeScan(final BluetoothDevice device, final int rssi,
                        final byte[] scanRecord) {
               ((Activity) mContext).runOnUiThread(new Runnable() {
                  @Override
                  public void run() {
                     IBeacon ib = new IBeacon();
                     if(device != null)
                        listener.beaconFound(ib.fromscanData(scanRecord,rssi));

                  }
               });
            }
         };
}
```

Figure 45: *BeaconScanner* (partially included) is responsible for all beacon operations.

```
. . .

public class MainActivity extends SampleActivityBase implements
AccelerometerListener, BeaconListener
{

   private BeaconScanner beaconScanner;

   . . .

   @Override
   public void beaconFound(IBeacon device)
   {
      Log.i(TAG, "Beacon : " + device.getRssi());
   }

   protected void onResume()
   {
      super.onResume();

      beaconScanner = new BeaconScanner(MainActivity.this,this);
      if(beaconScanner.isSupported())
      {
         beaconScanner.enableBLE();
         beaconScanner.scanLeDevice(true);
         Log.i(TAG, "Beacon scan started.");
      }
   }

   protected void onDestroy()
   {
      super.onDestroy();
      if(beaconScanner.isScanning())
      {
         beaconScanner.scanLeDevice(false);
      }

   }

   . . .

}
```

Figure 46: *MainActivity* (partially included) represents *WaistFacade* and uses *BeaconScanner*.

```
package com.clock;

import java.util.*;

import android.content.BroadcastReceiver;
import android.content.Context;
```

```
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Handler;
import android.os.SystemClock;
import android.text.format.Time;

public class Clock
{
   public static final int TICKPERSECOND=1;
   public static final int TICKPERMINUTE=0;

   private Time Time;
   private TimeZone TimeZone;
   private Handler Handler;
   private List<IClockListener> OnClockTickListenerList = new ArrayList<IClockListener>();

   private Runnable Ticker;

   private  BroadcastReceiver IntentReceiver;
   private IntentFilter IntentFilter;

   private int TickMethod=0;
   Context Context;

   SimulationManager sim_mngr = SimulationManager.getInstance();

   public Clock(Context context)
   {
      this(context, Clock.TICKPERMINUTE);
   }
   public Clock(Context context,int tickMethod)
   {
      this.Context=context;
      this.TickMethod=tickMethod;
      this.Time=new Time();
      this.Time.setToNow();

      switch (TickMethod)
      {
         case 0:
            this.StartTickPerSecond();
            break;
         case 1:
            this.StartTickPerMinute();
            break;

         default:
            break;
      }
   }
   private void Tick(long tickInMillis)
   {
      Clock.this.Time.set(Clock.this.Time.toMillis(true)+tickInMillis);
      this.NotifyOnTickListners();
   }
   private void NotifyOnTickListners()
   {
      switch (TickMethod)
```

```
        {
            case 0:
                for(IClockListener listner:OnClockTickListenerList)
                {
                    listner.OnSecondTick(Time);
                }
                break;
            case 1:
                for(IClockListener listner:OnClockTickListenerList)
                {
                    listner.OnMinuteTick(Time);
                }
                break;
        }
    }
    private void StartTickPerSecond()
    {
        this.Handler=new Handler();
        this.Ticker = new Runnable()
        {
            public void run()
            {
                Tick(200);
                long now = SystemClock.uptimeMillis();
                long next = now + 200;
                Handler.postAtTime(Ticker, next);
            }
        };
        this.Ticker.run();
    }
    public void StopTick()
    {
        if(this.IntentReceiver!=null)
        {
            this.Context.unregisterReceiver(this.IntentReceiver);
        }
        if(this.Handler!=null)
        {
            this.Handler.removeCallbacks(this.Ticker);
        }
    }
    public void AddClockTickListner(IClockListener listner)
    {
        this.OnClockTickListenerList.add(listner);

    }
}
```

Figure 47: *Clock* class.

```
package com.rule;

import com.algorithms.TSP;
import com.data.Destination;
import com.data.DestinationTypes;
import com.data.IRule;
```

```
import com.data.Purpose;
import com.data.Rule;
import com.data.RuleTypes;
import com.example.android.common.logger.Log;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by mustafa on 03.03.2016.
 */
public class RuleManager {
   private static RuleManager ourInstance = new RuleManager();

   public static RuleManager getInstance() {
      return ourInstance;
   }

   private RulesCollection rules_collection = RulesCollection.getInstance();

   public ArrayList<Destination>  resultList = null;

   private RuleManager()
   {
   }
   private ArrayList<Destination> Sort(List<IRule> list, Destination curentNode)
   {
      TSP tsp = new TSP();
      ArrayList<Destination> sortedList = new ArrayList<Destination>();
      ArrayList<Destination> tempList = new ArrayList<Destination>();
      List<IRule> sl;
      IRule start = curentNode;
      ArrayList<Destination> destList = new ArrayList<Destination>();
      ArrayList<Integer> destIndexList = new ArrayList<Integer>();
      Destination d;
      Destination dd;

      //find oredered destinations and their indices
      for(int i = 0;i<list.size();i++)
      {
         d = (Destination)list.get(i);
         if(d.destinationType == DestinationTypes.ORDERED_NODE_DESTINATION ||
               d.destinationType ==
DestinationTypes.ORDERED_NODE_AND_INFORMATION_DESTINATION)
         {
            destList.add(destList.size(), d);
            destIndexList.add(destIndexList.size(), i);
         }
      }
      //all destinations are unordered
      if(destList.size() == 0)
      {
         for(int i = 0;i<list.size();i++)
         {
            d = (Destination) list.get(i);
            sortedList.add(sortedList.size(), d);
         }
         sortedList = tsp.NNA(0, sortedList, curentNode, null);
```

```
        }
        //some are ordered
        else
        {
          //add first unordered destinations, if exist
          for(int i = 0;i<destIndexList.get(0);i++)
          {
            d = (Destination) list.get(i);
            sortedList.add(sortedList.size(), d);
          }

          //sort and add remaining blocks
          int destIndex = 0;
          boolean nextOrderedDestFound = false;
          for(int i = 0;i<destIndexList.size();i++)
          {
            tempList.clear();
            destIndex = destIndexList.get(i);
            dd = (Destination) list.get(destIndex);
            tempList.add(tempList.size(), dd);
            nextOrderedDestFound = false;
            for(int j = destIndex + 1;j<list.size() && !nextOrderedDestFound;j++)
            {
              d = (Destination) list.get(j);
              if(d.destinationType == DestinationTypes.ORDERED_NODE_DESTINATION ||
                    d.destinationType ==
DestinationTypes.ORDERED_NODE_AND_INFORMATION_DESTINATION)
              {
                nextOrderedDestFound = true;
              }
              else
                tempList.add(tempList.size(), d);
            }
            if(tempList.size()>1)
            {
              tempList = tsp.NNA(0, tempList, dd, null);
            }
            //add sub list to sortedList
            for(int k = 0;k<tempList.size();k++)
            {
              d = (Destination) tempList.get(k);
              sortedList.add(sortedList.size(), d);
            }
          }
        }
        return sortedList;
      }
      private List<IRule> PreExecute(Purpose pur, Destination curentNode)
      {
        List<IRule> list = null;
        List<IRule> sublist = null;
        boolean destinationListCompleted = false;
        boolean ruleFound = false;
        Rule rule;
        IRule irule = null;
        int indexOfRule = 0;

        do
```

```
        {
          // first evaluation
          if(list == null)
          {
             rule = rules_collection.GetRule(pur);
             list = rule.Execute();
          }
          // next evaluations
          else
          {
             // check if all destinations are found
             ruleFound = false;
             for(int i = 0;i<list.size() && !ruleFound;i++)
             {
                irule = list.get(i);
                if(irule.GetType() == RuleTypes.RULE)
                {
                   ruleFound = true;
                   indexOfRule = i;
                }
             }
             if(ruleFound)
                destinationListCompleted = false;
             else
                destinationListCompleted = true;

             // execute the found rule
             if(ruleFound)
             {
                rule = (Rule)irule;
                sublist = rule.Execute();

                //substitute rule with sublist
                list.remove(indexOfRule);
                for(int i = 0;i<sublist.size();i++)
                {
                   irule = (IRule)sublist.get(i);
                   list.add(indexOfRule + i, irule);
                }
             }
          }
        }
        while(!destinationListCompleted);
        return list;
     }
     public ArrayList<Destination> Execute(Purpose pur, Destination curentNode)
     {
        List<IRule> preList = PreExecute(pur, curentNode);
        ArrayList<Destination> list = new  ArrayList<Destination>();
        list = Sort(preList, curentNode);
        resultList = list;
        return list;
     }
  }
```

Figure 48: *RuleManager* singleton.

```
package com.system;

import android.util.Log;

import com.data.Destination;
import com.data.InteliEvaluateCommands;
import com.data.InteliState;
import com.data.InteliStateConditions;
import com.data.InteliStates;
import com.data.Purpose;
import com.data.PurposeConditions;
import com.data.PurposeTypes;
import com.data.StateConditions;
import com.data.States;
import com.rule.RuleManager;
import com.rule.RulesCollection;

import java.util.ArrayList;

/**
 * Created by mustafa on 16.03.2016.
 */
public class InteliEvaluationManager {
   private static InteliEvaluationManager ourInstance = new InteliEvaluationManager();

   public static InteliEvaluationManager getInstance() {
      return ourInstance;
   }

   private InteliStateManager intelistate_mngr = InteliStateManager.getInstance();
   private DestinationManager destination_mngr = DestinationManager.getInstance();
   private RuleManager rule_manager = RuleManager.getInstance();
   //private RulesCollection rules_collection = RulesCollection.getInstance();
   private PurposeManager purpose_mngr = PurposeManager.getInstance();
   ArrayList<Destination> result;

   private InteliEvaluationManager() {
   }

   String _resultstr = "";
   private InteliEvaluateCommands name;
   private ArrayList<Object> arguments;
   InteliState istate = new InteliState();

   public String Execute(InteliEvaluateCommands n, ArrayList<Object> args)
   {
      String resultstr = "";

      istate.stateType = InteliStates.EVALUATING;
      istate.stateCondition = InteliStateConditions.CONTINUE;
      arguments = args;
      name = n;

      new Thread(new Runnable() {
         @Override
         public void run()
         {
            //EVALUATION CONTINUE
```

```
            intelistate_mngr.SetState(istate);

            //TEST
            _resultstr = RunInBackground();

            //EVALUATION FINISHED
            istate = new InteliState();
            istate.stateType = InteliStates.EVALUATING;
            istate.stateCondition = InteliStateConditions.FINISHED;
            intelistate_mngr.SetState(istate);
          }
      }).start();
      resultstr = _resultstr;
      return resultstr;
   }

   private String RunInBackground()
   {
      String resultstr = "";

      if(name == InteliEvaluateCommands.EVALUATE_DESTINATIONS)
      {
         Purpose p = purpose_mngr.GetCurrentPurpose();
         if(p != null)
         {
            result = rule_manager.Execute(p, null);
            destination_mngr.SetDestinations(result);
      }
      return resultstr;
   }
}
```

Figure 49: *InteliEvaluationManager* singleton.

```
package com.system;

import android.os.SystemClock;

import com.data.EvaluateCommands;
import com.data.IBeacon;
import com.data.Node;
import com.data.Point;
import com.data.State;
import com.data.StateConditions;
import com.data.States;
import com.example.android.common.logger.Log;

import java.util.ArrayList;


public class EvaluationManager {
   private static EvaluationManager ourInstance = new EvaluationManager();

   private EvaluateCommands name;
   private ArrayList<Object> arguments;
   State state = new State();
```

```
    StateManager state_mngr;

    public static EvaluationManager getInstance() {
        return ourInstance;
    }
    ProgressOutOfAccesibilityPathManager progressaoutofaccpath_mngr =
ProgressOutOfAccesibilityPathManager.getInstance();
    private EvaluationManager() {
    }
    String _resultstr = "";

    public String Execute(EvaluateCommands n, ArrayList<Object> args)
    {
        String resultstr = "";
        state_mngr = StateManager.getInstance();

        state.stateType = States.EVALUATING;
        state.stateCondition = StateConditions.CONTINUE;
        arguments = args;
        name = n;

        new Thread(new Runnable() {
            @Override
            public void run()
            {
                //EVALUATION CONTINUE
                state_mngr.SetState(state);
                try
                {
                    _resultstr = RunInBackground();
                }
                catch (Exception e)
                {
                    //EVALUATION FINISHED
                    state.stateType = States.EVALUATING;
                    state.stateCondition = StateConditions.FINISHED;
                    state_mngr.SetState(state);
                }
                //EVALUATION FINISHED
                state.stateType = States.EVALUATING;
                state.stateCondition = StateConditions.FINISHED;
                state_mngr.SetState(state);
            }
        }).start();
        resultstr = _resultstr;
        return resultstr;
    }

    private String RunInBackground()
    {
        String resultstr = "";
        PathManager path_mngr = PathManager.getInstance();

        if(name == EvaluateCommands.FIND_SHORTEST_PATH_BETWEEN_TWO_POINTS)
        {
            Node startNode = (Node)arguments.get(0);
            Node destNode = (Node)arguments.get(1);
            path_mngr.FindShortestPath(0,startNode,destNode);
```

```
            path_mngr.FillProgressManager();
            Log.d("XXXXXXXXXX:", "
FIND_SHORTEST_PATH_BETWEEN_TWO_POINTS ");
        }
        else if(name ==
EvaluateCommands.HEADING_IS_WRONG_FIND_CORRECT_HEADING)
        {
            ProgressHeadingManager proghead_mngr = ProgressHeadingManager.getInstance();
            int reqHeading = (int) arguments.get(0);
            int actHeading = (int) arguments.get(1);
            proghead_mngr.EvaluateHeadingInstruction(reqHeading, actHeading);
            Log.d("XXXXXXXXXX:", "
HEADING_IS_WRONG_FIND_CORRECT_HEADING ");
        }
        else if(name == EvaluateCommands.UPDATE_PROGRESS_MANAGER)
        {
            ProgressManager prog_mngr = ProgressManager.getInstance();
            PositionManager position_mngr = PositionManager.getInstance();
            Node curNode = (Node) arguments.get(0);
            //get current node
            Node nrfid = position_mngr.GetPositionNodeRfid();
            prog_mngr.UpdateProgressManagerGivenCurrentNode(nrfid);
            Log.d("XXXXXXXXXX:", " UPDATE_PROGRESS_MANAGER ");
        }
        else if(name ==
EvaluateCommands.USER_IS_NOT_ON_ACCESSIBILITY_PATH_DIRECT_USER_TO_ACC
ESSIBILITY_PATH)
        {
            Point curBeaconPoint = (Point) arguments.get(0);
            Node curNode = (Node) arguments.get(1);
            int actHeading = (int) arguments.get(2);
            Log.d("--------BEACON POSITION ----", "X:" + curBeaconPoint.X + ", Y:" +
curBeaconPoint.Y + "------------------");
progressaoutofaccpath_mngr.EvaluateIBeaconPositionInstruction(curBeaconPoint, curNode,
actHeading);
            resultstr = progressaoutofaccpath_mngr.instruction_ibeacon_position;
            Log.d("XXXXXXXXXX:", "
USER_IS_NOT_ON_ACCESSIBILITY_PATH_DIRECT_USER_TO_ACCESSIBILITY_P
ATH ");
        }
        else if(name ==
EvaluateCommands.USER_IS_NOT_ON_ACCESSIBILITY_PATH_AND_POSITION_CAN_N
OT_BE_DETERMINED)
        {
            progressaoutofaccpath_mngr.EvaluateNoPositionInstruction();

            Log.d("XXXXXXXXXX:", "
USER_IS_NOT_ON_ACCESSIBILITY_PATH_AND_POSITION_CAN_NOT_BE_DETER
MINED ");
        }
        //SystemClock.sleep(8000);
        return resultstr;
    }
}
```

Figure 50: *EvaluationManager* singleton.

```
package com.system;

import com.configuration.UserInputManagerConfig;
import com.data.Destination;
import com.data.InteliEvaluateCommands;
import com.data.InteliState;
import com.data.InteliStateConditions;
import com.data.InteliStates;
import com.data.State;
import com.data.StateConditions;
import com.data.States;
import com.example.android.common.logger.Log;

import java.util.Date;

public class InteliManager {
   private static InteliManager ourInstance = new InteliManager();

   public static InteliManager getInstance() {
      return ourInstance;
   }
   private InteliStateManager intelistate_mngr = InteliStateManager.getInstance();
   private InteliEvaluationManager intelievaluate_mngr = InteliEvaluationManager.getInstance();
   private UserInputManager userinput_mngr = UserInputManager.getInstance();
   private Manager mngr = Manager.getInstance();
   private DestinationManager destination_mngr = DestinationManager.getInstance();
   private StateManager state_mngr = StateManager.getInstance();
   private PurposeManager purpose_mngr = PurposeManager.getInstance();

   private InteliManager() {
   }
   public void Execute()
   {
      //Get State
      InteliState istate = intelistate_mngr.GetState();

      //EVALUATE
      if(istate.stateType == InteliStates.EVALUATING && istate.stateCondition ==
InteliStateConditions.STARTED)
      {
         intelievaluate_mngr.Execute(InteliEvaluateCommands.EVALUATE_DESTINATIONS,
null);
      }
      //WAIT
      else if(istate.stateType == InteliStates.WAITING && istate.stateCondition ==
InteliStateConditions.CONTINUE)
      {
         boolean managerIsReadyForDestination = false;
         boolean destinationApprovedByUser = false;
         boolean destinationListFinished = false;

         //CHECK IF MANAGER IS READY----------------------------------------------------------
         State state = state_mngr.GetState();
         if(state.stateType == States.IDLE && state.stateCondition ==
StateConditions.CONTINUE)
            managerIsReadyForDestination = true;
         else
            managerIsReadyForDestination = false;
```

```
        //CHECK IF MANAGER IS READY----------------------------------------------------------
        //CHECK IF DESTINATION LIST IS EMPTY-------------------------------------------------
        Destination dest = destination_mngr.GetDestination();
        //Destination dest = destination_mngr.GetCurrentDestination();
        if(dest == null)
           destinationListFinished = true;
        else
           destinationListFinished = false;
        //CHECK IF DESTINATION LIST IS EMPTY-------------------------------------------------
        //CHECK DESTINATION APPROVAL---------------------------------------------------------
        if(!destinationListFinished)
        {
           if (userinput_mngr.DestinationApproved)
           {
              userinput_mngr.DestinationApproved = false;
              destinationApprovedByUser = true;
           } else if (!destinationApprovedByUser && managerIsReadyForDestination) {
              //SEND MESSAGE TO USER, CHECK LAST SENT DATE
              Date lastmsgsentdate = userinput_mngr.lastMsgSentDate;

              if (lastmsgsentdate == null)
              {
                 SendMessage(dest.destinationInfo + " gitmek için hedefi onayla butonuna
bas");
              } else
              {
                 Date currentDate = new Date();
                 int diffInSec = (int) Math.abs(currentDate.getTime() - lastmsgsentdate.getTime()) /
1000;
                 if (diffInSec > UserInputManagerConfig.DURATION_IN_SEC_FOR_MESSAGE)
                 {
                    SendMessage(dest.destinationInfo + " gitmek için hedefi onayla butonuna
bas");
                 }
              }
           }
        }
        //CHECK DESTINATION APPROVAL---------------------------------------------------------
        //SYSTEM IS READY FOR NEXT
DESTINATION**************************************************
        if(managerIsReadyForDestination && destinationApprovedByUser &&
!destinationListFinished)
        {

           //MANAGER IDLE FINISHES
           State s = new State();
           s.stateType = States.IDLE;
           s.stateCondition = StateConditions.FINISHED;
           state_mngr.SetState(s);
        }
        //SYSTEM IS READY FOR NEXT
DESTINATION**************************************************
        //DESTINATION LIST
FINISHED*****************************************************************
        else if(destinationListFinished)
        {
           State sc = state_mngr.GetState();
```

```
            if(sc.stateType == States.IDLE && sc.stateCondition ==
StateConditions.CONTINUE)
            {
               purpose_mngr.currentPurpose = null;
               SendMessage("rota tamamlandı");
               //INTELIMANAGER IDLE STARTS
               InteliState is = new InteliState();
               is.stateType = InteliStates.IDLE;
               is.stateCondition = InteliStateConditions.STARTED;
               intelistate_mngr.SetState(is);
               mngr.InitializeSetup();
            }
         }
      }
   }
   private void SendMessage(String s)
   {
      mngr.SendMessage(s);
      userinput_mngr.lastMsgSentDate = new Date();
   }
}
```

Figure 51: *InteliManager* singleton.

```
package com.system;

import com.algorithms.BeaconsAccessibilityPathDirectionOperations;
import com.data.Destination;
import com.data.EvaluateCommands;
import com.data.Graph;
import com.data.Node;
import com.data.Orientation;
import com.data.Point;
import com.data.Progress;
import com.data.State;
import com.data.StateConditions;
import com.data.States;
import com.example.android.common.logger.Log;
import com.example.android.simulasyon.SimulationManager;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;

public class Manager {

   private static Manager ourInstance = new Manager();
   public static Manager getInstance() {
      return ourInstance;
   }

   EvaluationManager eval_mngr = EvaluationManager.getInstance();
   StateManager state_mngr = StateManager.getInstance();
   OrientationManager orientation_mngr = OrientationManager.getInstance();
   BuildingGraphManager building_mngr = BuildingGraphManager.getInstance();
   ProgressManager progress_manager = ProgressManager.getInstance();
```

```
   ProgressOutOfAccesibilityPathManager progressaoutofaccpath_mngr =
ProgressOutOfAccesibilityPathManager.getInstance();
   PositionManager position_mngr = PositionManager.getInstance();
   SimulationManager sim_mngr = SimulationManager.getInstance();
   DestinationManager destination_mngr = DestinationManager.getInstance();
   BipShakeManager bipshake_mngr = BipShakeManager.getInstance();

   private IManagerListener listener;
   private String userInput = "";

   boolean userIsOnAccessibilityPath = false;
   boolean currentNodeIsOk = false;
   boolean userOrientationIsOk = false;
   boolean currentNodeIsNotOkButInShortestPath = false;
   boolean userPositionTriedToBeDeterminedByIbeacon = false;
   boolean userPositionCanBeFoundByIBeacon = false;

   BeaconsAccessibilityPathDirectionOperations beaaccesspathdiroperations = new
BeaconsAccessibilityPathDirectionOperations();
   private Manager()
   {
   }
   public void SetListener(IManagerListener l)
   {
      listener = l;
   }
   public void InitializeSetup()
   {
      //clear progress_manager
      progress_manager.ClearList();
   }
   String lastState = "";
   int lastNodeId = -1;
   String lastInstr = "";
   Node prevNode = null;
   Node nextNode = null;
   Node outOfRouteNode = null;
   Node outOfRouteNextNode = null;
   Date outOfRouteNodeMsgDate = null;
   boolean oneNodeNear;
   boolean evaluatingStartedStartNodeCanNotBeDetermined = false;
   String lastMessage = "";
   public void Execute()
   {
      Orientation lastOrientation;
      EvaluateCommands command;
      ArrayList<Object> command_args = new ArrayList<Object>();
      //Get State
      State state = state_mngr.GetState();
      //not on accessibility path
      if(!bipshake_mngr.userIsOnAccessibilityPath)
      {
         Date now = new Date();
         int diffInSec = (int) Math.abs(now.getTime() -
bipshake_mngr.lastTagReadDate.getTime());
         int diffInSecShake = (int) Math.abs(now.getTime() -
bipshake_mngr.lastShakeDate.getTime());
         if (diffInSec > BipShakeManager.DURATION_IN_MILISEC_TO_BEEP &&
```

```
                diffInSecShake >
BipShakeManager.DURATION_IN_MILISEC_TO_SEND_SHAKE_SIGNAL)
            {
                bipshake_mngr.lastShakeDate = new Date();
                SendMessage("pozisyon belirlenemiyor", "", "", "", "", "0", "", "", null);
            }
        }
        //Evaluate
        if((state.stateType == States.EVALUATING && state.stateCondition ==
StateConditions.STARTED) ||
            (state.stateType == States.EVALUATING && state.stateCondition ==
StateConditions.CONTINUE && evaluatingStartedStartNodeCanNotBeDetermined))
        {
            //userInput = input_mngr.GetCurrentInput();
            //if(userInput.compareTo("hedefe") == 0)
            Destination d = destination_mngr.GetDestination();
            if(d != null)
            {
                //clear
                progress_manager.ClearList();
                //find shortest path and fill ProgressManager
                command =
EvaluateCommands.FIND_SHORTEST_PATH_BETWEEN_TWO_POINTS;
                Graph graph = building_mngr.GetGraph(0);
                String pr = "" + 0;
                lastOrientation = orientation_mngr.GetLastOrientation();
                if (lastOrientation != null)
                {
                    pr = "" + ((int) lastOrientation.GetHeading() + 45);
                }
                Node startNode;
                if(!sim_mngr.open_rfid)
                    startNode = sim_mngr.GetStartNode();
                else
                    startNode = position_mngr.GetPositionNodeRfid();
                if(startNode != null)
                {
                    destination_mngr.IncreaseCounter();
                    Node destNode = graph.getVertex(d.destinationNodeId);
                    command_args.clear();
                    command_args.add(command_args.size(), startNode);
                    command_args.add(command_args.size(), destNode);
                    eval_mngr.Execute(command, command_args);
                    String sx = "";
                    String sy = "";
                    sx = "" + startNode.getX();
                    sy = "" + startNode.getY();
                    SendMessage("hedefe yönlendiriliyorsun", sx, sy, sx, sy, pr, "", "", null);
                    evaluatingStartedStartNodeCanNotBeDetermined = false;
                }
                else
                {
                    SendMessage("pozisyon belirlenemiyor", "", "", "", "", pr, "", "", null);
                    evaluatingStartedStartNodeCanNotBeDetermined = true;
                }
            }
        }
        //Monitor
```

```
      else if(state.stateType == States.MONITORING && state.stateCondition ==
StateConditions.CONTINUE)
      {
         String insttouser = "";
         String optionstr = "";

         //Get Progress Info
         Progress p = progress_manager.GetNextProgress();

         if(p != null && !p.finished)
         {
            //Execute
Progress*************************************************************************
            //Find actual position, RFID----------------------------------------------------
            Node nrfid;
            //TEST TEST TEST ********************************
            if(sim_mngr.open_rfid)
               nrfid = position_mngr.GetPositionNodeRfid();
            else
            {
               nrfid = p.node;
            }
            if (nrfid != null) {
               Log.d("Manager received RFID node:", nrfid.GetIRfidInfo());
               Log.d("Progress node:", p.node.GetIRfidInfo());
            }
            //Find position beacon----------------------------------------------------------
            Point pBeacon = null;
            Point pBeacon_sent = null;
            if(sim_mngr.open_beacon)
               pBeacon = position_mngr.GetPositionPointBeacon();
            if(pBeacon != null)
            {
               Log.d("--------BEACON POSITION", "X:" + pBeacon.X + ", Y:" + pBeacon.Y +
"------------------");
               //beacon_mngr.PrintListLog();
            }
            //Find actual heading
            lastOrientation = orientation_mngr.GetLastOrientation();
            //userIsOnAccessibilityPath
            if(nrfid != null)
            {
               userIsOnAccessibilityPath = true;
               if(prevNode == null || nextNode == null)
               {
                  prevNode = nextNode;
                  nextNode = nrfid;
               }
               else if(nrfid.getID() != nextNode.getID())
               {
                  prevNode = nextNode;
                  nextNode = nrfid;
               }
            }
            else
               userIsOnAccessibilityPath = false;

            //userOrientationIsOk
```

```
            if(Math.abs(p.heading - lastOrientation.GetHeading()) <= 60)
               userOrientationIsOk = true;
            else
               userOrientationIsOk = false;

            //TEST TEST TEST ********************************
            if(!sim_mngr.open_orientation)
               userOrientationIsOk = true;
            //USER IS ON ACCESSIBILITY PATH
            if(userIsOnAccessibilityPath)
            {
               //currentNodeIsOk
               if(p.node.equals(nrfid))
                  currentNodeIsOk = true;
               else
                  currentNodeIsOk = false;

               //USER IS ON CORRECT NODE
               if(currentNodeIsOk)
               {
                  outOfRouteNode = null;
                  outOfRouteNodeMsgDate = null;
                  //USER ORIENTATION IS
CORRECT************************************************
                  if(userOrientationIsOk)
                  {
                     lastNodeId = p.node.getID();
                     optionstr = "correct_node_correct_orientation";
                     lastState = optionstr;
                     //sonraki progrese geçmesi için
                     p.finished = true;
progress_manager.AddProgressAtPosition(p.index, p);
                     insttouser = p.GetOutputMsg();
                     lastInstr = insttouser;
                     lastMessage = insttouser;
                     SendMessageSimulation(p.GetOutputMsg());
                  }
                  //USER ORIENTATION IS NOT
CORRECT********************************************
                  else
                  {
                     optionstr = "correct_node_wrong_orientation";
                     lastState = optionstr;
                     ProgressHeadingManager proghead_mngr =
ProgressHeadingManager.getInstance();
                     if(proghead_mngr.headingEvaluated)
                     {
                        //SendMessage(proghead_mngr.instruction);
                        insttouser = proghead_mngr.instruction;
                        SendMessageSimulation(proghead_mngr.instruction);
                        proghead_mngr.headingEvaluated = false;
                     }
                     else
                     {
                        State ss = new State();
                        ss.stateType = States.EVALUATING;
                        ss.stateCondition = StateConditions.STARTED;
                        state_mngr.SetState(ss);
```

```
                    command =
EvaluateCommands.HEADING_IS_WRONG_FIND_CORRECT_HEADING;
                    command_args.clear();
                    command_args.add(command_args.size(), p.heading);
                    command_args.add(command_args.size(), (int) lastOrientation.GetHeading());
                    eval_mngr.Execute(command, command_args);
                  }
                }
              }
              //USER IS ON WRONG NODE
              else if(!(lastNodeId == nrfid.getID()) || (lastNodeId == nrfid.getID() &&
!lastState.equals("correct_node_correct_orientation")))
              {
                //currentNodeIsNotOkButInShortestPath
                if(progress_manager.IsNodeOnShortestPath(nrfid))
                  currentNodeIsNotOkButInShortestPath = true;
                else
                  currentNodeIsNotOkButInShortestPath = false;

                //USER IS ON A NODE WHICH IS ON THE SHORTEST PATH
                if(currentNodeIsNotOkButInShortestPath)
                {
                  outOfRouteNode = null;
                  outOfRouteNodeMsgDate = null;
                  optionstr = "wrong_node_on_shortest_path";
                  lastState = optionstr;
                  State ss = new State();
                  ss.stateType = States.EVALUATING;
                  ss.stateCondition = StateConditions.STARTED;
                  state_mngr.SetState(ss);
                  command = EvaluateCommands.UPDATE_PROGRESS_MANAGER;
                  command_args.clear();
                  command_args.add(command_args.size(), nrfid);
                  eval_mngr.Execute(command, command_args);
                }
                //USER IS ON A NODE WHICH IS NOT ON SHORTEST
PATH**************************
                else
                {
                  oneNodeNear = false;
                  //user pass the point by only one node
                  if(outOfRouteNode == null)
                  {
                    outOfRouteNode = prevNode;
                    outOfRouteNextNode = nextNode;
                  }
                  oneNodeNear = building_mngr.DirectEdgeBetweenNodes(outOfRouteNode,
nrfid);
                  if(!oneNodeNear)
                  {
                    outOfRouteNode = null;
                    outOfRouteNodeMsgDate = null;
                    //user pass the point by more than one points
                    optionstr = "wrong_node_not_on_shortest_path";
                    lastState = optionstr;
                    //find shortest path and fill ProgressManager
                    command =
EvaluateCommands.FIND_SHORTEST_PATH_BETWEEN_TWO_POINTS;
```

```
                    progress_manager.ClearList();

                    Graph graph = building_mngr.GetGraph(0);
                    destination_mngr.DecreaseCounter();
                    Destination d = destination_mngr.GetDestination();
                    destination_mngr.IncreaseCounter();
                    Node destNode = graph.getVertex(d.destinationNodeId);

                    d.ShowDestination();

                    if (destNode == null) {
                       insttouser = "rotadan çıktın hedef yok";
                    } else {
                       command_args.clear();
                       command_args.add(command_args.size(), nrfid);
                       command_args.add(command_args.size(), destNode);
                       eval_mngr.Execute(command, command_args);
                       insttouser = "rotadan çıktın hedef tekrar hesaplanıyor";
                    }
                 }
                 else
                 {
                    if(outOfRouteNodeMsgDate == null)
                    {
                       outOfRouteNodeMsgDate = new Date();
                       insttouser = "rotadan çıkıyorsun";
                       if(!insttouser.equals(""))
                       {
                          insttouser = "rotadan çıkıyorsun son mesajın " +  lastMessage;
                       }
                    }
                    else
                    {
                       Date now = new Date();
                       int diffInSec = (int)Math.abs(now.getTime() -
outOfRouteNodeMsgDate.getTime());
                       if(diffInSec>6000)
                       {
                          outOfRouteNodeMsgDate = new Date();
                          insttouser = "rotadan çıkıyorsun";
                          if(!insttouser.equals(""))
                          {
                             insttouser = "rotadan çıkıyorsun son mesajın " +  lastMessage;
                          }
                       }
                    }
                 }
              }
           }
        }
        //USER IS OUT OF ACCESSIBILITY PATH
        else
        {
           //userPositionCanBeFoundByIBeacon
           if(pBeacon != null)
              userPositionCanBeFoundByIBeacon = true;
           else
              userPositionCanBeFoundByIBeacon = false;
```

```
            //USER POSITION CAN BE DETERMINED BY IBEACON
            if (userPositionCanBeFoundByIBeacon)
            {
               optionstr = "out_of_path_position_determined_by_ibeacon";
               lastState = optionstr;
               State ss = new State();
               ss.stateType = States.EVALUATING;
               ss.stateCondition = StateConditions.STARTED;
               state_mngr.SetState(ss);
               command =
EvaluateCommands.USER_IS_NOT_ON_ACCESSIBILITY_PATH_DIRECT_USER_TO_ACC
ESSIBILITY_PATH;
               command_args.clear();
               pBeacon_sent = pBeacon;
               command_args.add(command_args.size(), pBeacon);
               command_args.add(command_args.size(), null);
               command_args.add(command_args.size(), (int) lastOrientation.GetHeading());
               insttouser = eval_mngr.Execute(command, command_args);
               SendMessageSimulation(insttouser);
            }

            else
            {
               optionstr = "out_of_path_position_can_not_determined_by_ibeacon";
               lastState = optionstr;
               State ss = new State();
               ss.stateType = States.EVALUATING;
               ss.stateCondition = StateConditions.STARTED;
               state_mngr.SetState(ss);
               command =
EvaluateCommands.USER_IS_NOT_ON_ACCESSIBILITY_PATH_AND_POSITION_CAN_N
OT_BE_DETERMINED;
               eval_mngr.Execute(command, null);
SendMessageSimulation(progressaoutofaccpath_mngr.instruction_no_position);
               insttouser = progressaoutofaccpath_mngr.instruction_no_position;
            }
         }
         if(nrfid != null) {
            progress_manager.PrintProgress();
         }

         try
         {
            if(insttouser != null && insttouser != "")
            {
               String sx = "";
               String sy = "";
               String pr = "" + ((int) lastOrientation.GetHeading() + 45);
               if(!sim_mngr.open_orientation)
                  pr = "" + (p.heading + 45);

               //BEACON FOUND POSITION
               String bx = "";
               String by = "";
               Point pb = progressaoutofaccpath_mngr.beaconPoint;
               if(pb != null)
               {
```

```
                bx = "" + Math.abs((int)pb.X);
                by = "" + Math.abs((int)pb.Y);
            }

            //BEACONS
            List<Point> blist = position_mngr.GetBeaconPositions();

            try
            {
                sx = "" + nrfid.getX();
                sy = "" + nrfid.getY();
            }
            catch (Exception e)
            {
                sx = "";
                sy = "";
            }

            if(optionstr == "correct_node_correct_orientation")
            {
                SendMessage(insttouser, sx, sy, sx, sy, pr, bx, by, blist);
            }
            else if(optionstr == "correct_node_wrong_orientation")
            {
                SendMessage(insttouser, sx, sy, sx, sy, pr, bx, by, blist);
            }
            else if(optionstr == "wrong_node_on_shortest_path")
            {
                SendMessage(insttouser, sx, sy, sx, sy, pr, bx, by, blist);
            }
            else if(optionstr == "wrong_node_not_on_shortest_path")
            {
                SendMessage(insttouser, sx, sy, sx, sy, pr, bx, by, blist);
            }
            else if(optionstr == "out_of_path_position_determined_by_ibeacon")
            {
                sx = bx;
                sy = by;
                SendMessage(insttouser, sx, sy, sx, sy, pr, bx, by, blist);
            }
            else if(optionstr == "out_of_path_position_can_not_determined_by_ibeacon")
            {
                SendMessage(insttouser, sx, sy, sx, sy, pr, bx, by, blist);
            }
            else
            {
                SendMessage(insttouser, sx, sy, sx, sy, pr, "", "", null);
            }
        }
    }
    catch (Exception e)
    {

    }
}
else
{
    State ss = new State();
```

```
            ss.stateType = States.MONITORING;
            ss.stateCondition = StateConditions.FINISHED;
            state_mngr.SetState(ss);
         }
      }
   }
   private void SendMessage(String ins, String scollx, String scrolly, String px, String py, String
prot,
                     String bpX, String bpY, List<Point> beaconList)
   {
      if(listener!=null)
      {
         //send message to client application
         String sendmsgdivider = "|";
         String sendmsg = "";
         String sendmsg_instr = ins;
         String sendmeg_pointerrotate = prot;
         String sendmeg_scrollx = scollx;
         String sendmeg_scrolly = scrolly;
         String sendmeg_poinerx = px;
         String sendmeg_pointery = py;
         sendmsg = sendmsg_instr + sendmsgdivider +
               sendmeg_scrollx + sendmsgdivider +
               sendmeg_scrolly + sendmsgdivider +
               sendmeg_poinerx + sendmsgdivider +
               sendmeg_pointery + sendmsgdivider +
               sendmeg_pointerrotate + sendmsgdivider;

         if(!bpX.equals("") && !bpY.equals(""))
         {
            sendmsg += bpX + sendmsgdivider + bpY + sendmsgdivider;
            if(beaconList != null)
            {
               if(beaconList.size()>0)
               {
                  Point p;
                  for(int i = 0;i<beaconList.size();i++)
                  {
                     p = beaconList.get(i);
                     sendmsg += ((int)p.X) + sendmsgdivider + ((int)p.Y) + sendmsgdivider;
                  }
               }
            }
         }
         listener.ManagerSendsMessage(sendmsg);
      }
   }
   public void SendMessage(String msg)
   {
      if(listener!=null)
      {
         listener.ManagerSendsMessage(msg);
      }
   }
}
```

Figure 52: *Manager* singleton.

Appendix B: The Survey

Kullanıcı Bilgileri Anketi (test öncesi)

1. **Cinsiyetiniz**:
    a. Erkek
    b. Kadın
2. **Yaşınız:**
    a. 18 - 25
    b. 26 - 39
    c. 40 - 59
    d. >60
3. **Eğitim durumunuz:**
    a. Yok
    b. İlkokul
    c. Ortaokul
    d. Lise
    e. Üniversite
    f. Yüksek öğrenim
4. **Görme engelli bastonunu kullanmayı ne derece biliyorsunuz?** Lütfen 1 den 5 kadar derecelendiriniz.
    a. 5 (çok iyi)
    b. 4 (iyi)
    c. 3 (orta)
    d. 2 (kötü)
    e. 1 (çok kötü)
5. **Evinizden dışarıya ne sıklıkla çıkarsınız?**
    a. Çok sık, (genellikle her gün)
    b. Sıklıkla (haftada iki – dört gün)
    c. Orta sıklıkta (haftada bir – iki gün)
    d. Nadiren (ayda 1 – 2 gün)

e. Çok nadiren (çok zorunlu olduğunda, senede birkaç gün)

6. **Evinizden dışarıya çıkma nedeniniz sıralayınız:**

a. Eğitim: …….

b. Sağlık: …….

c. İş:……..

d. Alış-veriş:……

e. Sosyal:…….

f. Diğer 1(……): …….

g. Diğer 2(……): …….

h. Diğer 3(……): …….

7. **Akıllı telefon veya tablet kullanma sıklığınız nedir?**

a. Çok sık, (genellikle her gün)

b. Sıklıkla (haftada iki – dört gün)

c. Orta sıklıkta (haftada bir – iki gün)

d. Nadiren (ayda 1 – 2 gün)

e. Çok nadiren (çok gerekli olduğunda, senede birkaç gün)

8. **Akıllı telefon veya tabletlerdeki talkback gibi erişilebilirlik yazılımlarını kullanma başarınız nedir?** Lütfen 1 den 5 kadar derecelendiriniz.

a. 5 (çok iyi)

f. 4 (iyi)

g. 3 (orta)

h. 2 (kötü)

i. 1 (çok kötü)

Görev Bilgileri ve Gözlemler (test)

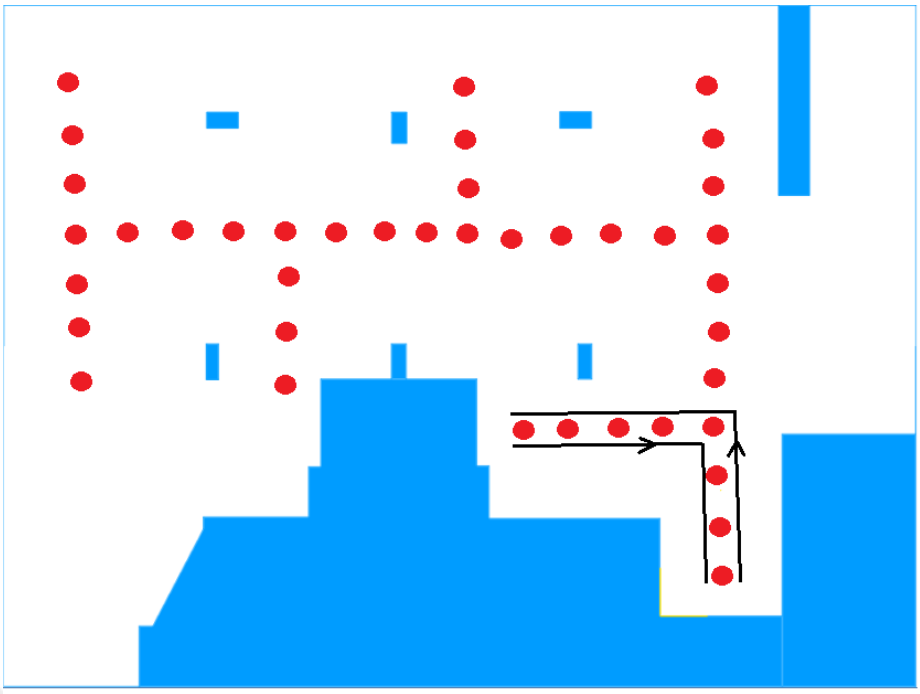
Görev 1: **"Çıkış" noktasındasınız amaç olarak "hasta kabul" seçip yönlendirme komutları ile ilerleyiniz. "Hasta kabul" e ulaştıktan sonra da "Çıkış" noktasına geri dönünüz.**

Görevdeki toplam hedef sayısı: 1

Görev toplam uzunluğu (metre): 14

Görevdeki toplam dönme noktası sayısı: 2

1. Ara **yüzü kullanmada** ne derece başarılı oldu?
    a. Yardım almadan kullanabildi
    b. Bir kere yardım alarak kullanabildi
    c. İki – üç defa yardım alarak kullanabildi
    d. Üçten fazla yardım alarak kullanabildi
    e. Hiç kullanamadı
2. Ara yüzü kullanarak **amaç belirlemesi ne kadar sürdü** (saniye):……..
3. **Yönlendirme komutlarını ne derece doğru** uyguladı?
    a. Hiç şaşırmadan komuta göre hareket etti
    b. Bir kere şaşırdı
    c. İki – Üç defa şaşırdı
    d. Üçten fazla şaşırdı
    e. Komutların hiçbirini doğru uygulayamadı
4. Kaç kere **erişilebilirlik yolu dışına çıktı**:…….
5. **Erişilebilirlik yolu dışına çıktığı durumların kaçında sistemi kullanarak tekrar yola girebildi**:……..
6. Kaç kere **erişilebilirlik yolu üzerinde rota dışına çıktı**:…….
7. Görevi tamamlama **hızı** (saniye):……
8. Hedeflerin **kaçına ulaşabildi**:…….
9. Kullanıcının **hareketleri**:



Görev 2: **Bulunduğunuz "Çıkış" noktasından "Muayene – Dahiliye" amacını seçip ilerleyiniz.**

Görevdeki toplam hedef sayısı: 3

Görev toplam uzunluğu (metre): 37

Görevdeki toplam dönme noktası sayısı: 4

10. Ara **yüzü kullanmada** ne derece başarılı oldu?
    a. Yardım almadan kullanabildi
    b. Bir kere yardım alarak kullanabildi
    c. İki – üç defa yardım alarak kullanabildi
    d. Üçten fazla yardım alarak kullanabildi
    e. Hiç kullanamadı
11. Ara yüzü kullanarak **amaç belirlemesi ne kadar sürdü** (saniye):……..
12. **Yönlendirme komutlarını ne derece doğru** uyguladı?
    a. Hiç şaşırmadan komuta göre hareket etti
    b. Bir kere şaşırdı
    c. İki – Üç defa şaşırdı

d. Üçten fazla şaşırdı

e. Komutların hiçbirini doğru uygulayamadı

13. Kaç kere **erişilebilirlik yolu dışına çıktı**:……

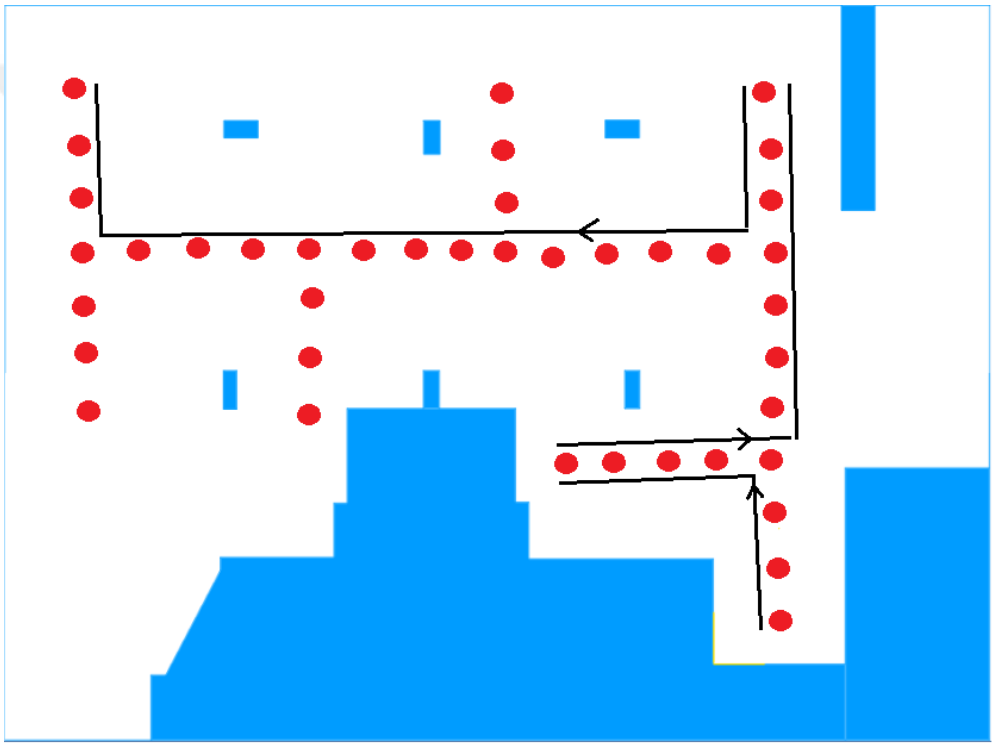
14. **Erişilebilirlik yolu dışına çıktığı durumların kaçında sistemi kullanarak tekrar yola girebildi**:……..

15. Kaç kere **erişilebilirlik yolu üzerinde rota dışına çıktı**:…….

16. Görevi tamamlama **hızı** (saniye):……

17. Hedeflerin **kaçına ulaşabildi**:…….

18. Kullanıcının **hareketleri**:



Görev 3: **Bulunduğunuz "Muayene – Dahiliye" noktasından "tahlil - kan" amacını seçip ilerleyiniz.**

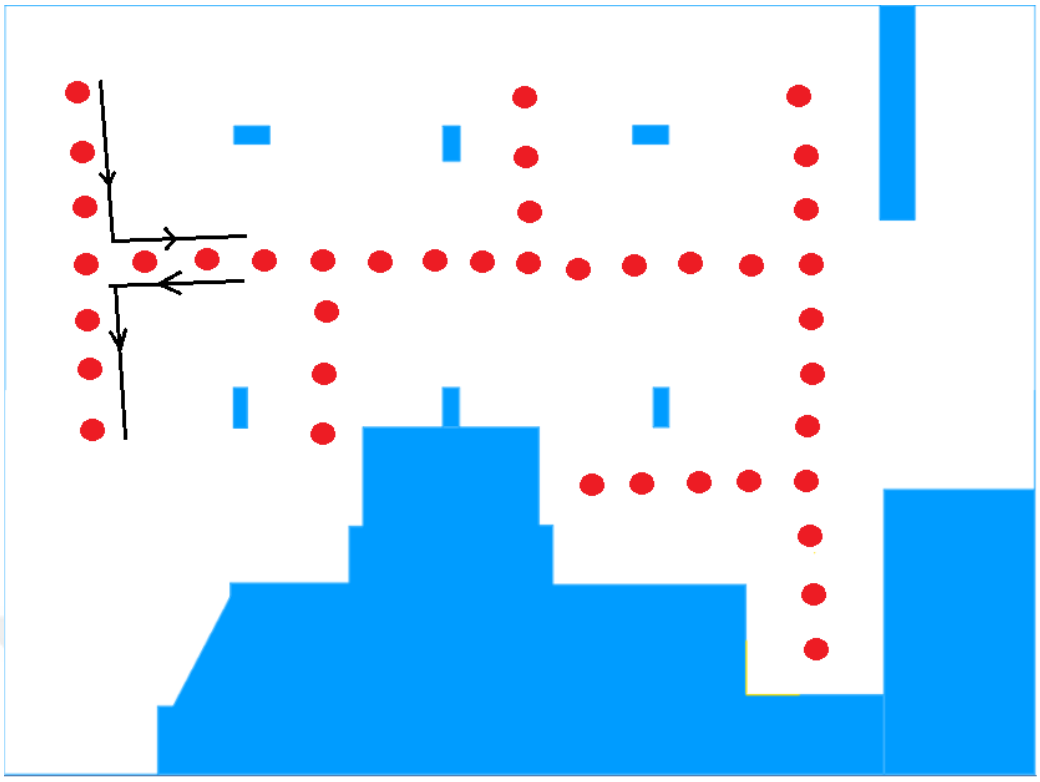
(Burada kullanıcı belirlenen noktada erişilebilirlik yolu üzerindeki yanlış bir yere gidecek ve sistemin tekrar yol hesaplaması yapması sağlanacaktır)

Görevdeki toplam hedef sayısı: 1

Görev toplam uzunluğu (metre): 7 (yanlış yol mesafesi hariç)

Görevdeki toplam dönme noktası sayısı: -

19. Ara **yüzü kullanmada** ne derece başarılı oldu?
    a. Yardım almadan kullanabildi
    b. Bir kere yardım alarak kullanabildi
    c. İki – üç defa yardım alarak kullanabildi
    d. Üçten fazla yardım alarak kullanabildi
    e. Hiç kullanamadı
20. Ara yüzü kullanarak **amaç belirlemesi ne kadar sürdü** (saniye):……..
21. **Yönlendirme komutlarını ne derece doğru** uyguladı?
    a. Hiç şaşırmadan komuta göre hareket etti
    b. Bir kere şaşırdı
    c. İki – Üç defa şaşırdı
    d. Üçten fazla şaşırdı
    e. Komutların hiçbirini doğru uygulayamadı
22. Kaç kere **erişilebilirlik yolu dışına çıktı**:……
23. **Erişilebilirlik yolu dışına çıktığı durumların kaçında sistemi kullanarak tekrar yola girebildi**:……..
24. Kaç kere **erişilebilirlik yolu üzerinde rota dışına çıktı**:…….
25. Görevi tamamlama **hızı** (saniye):……
26. Hedeflerin **kaçına ulaşabildi**:…….
27. Kullanıcının **hareketleri**:



Görev 4: **Bulunduğunuz "tahlil - kan" noktasından "çıkış" amacını seçip ilerleyiniz.**

(Belirlenen noktada kullanıcı erişilebilirlik yolundan çıkartılacak ve sistemi takip ederek tekrar erişilebilirlik yoluna girmesi istenecektir.)

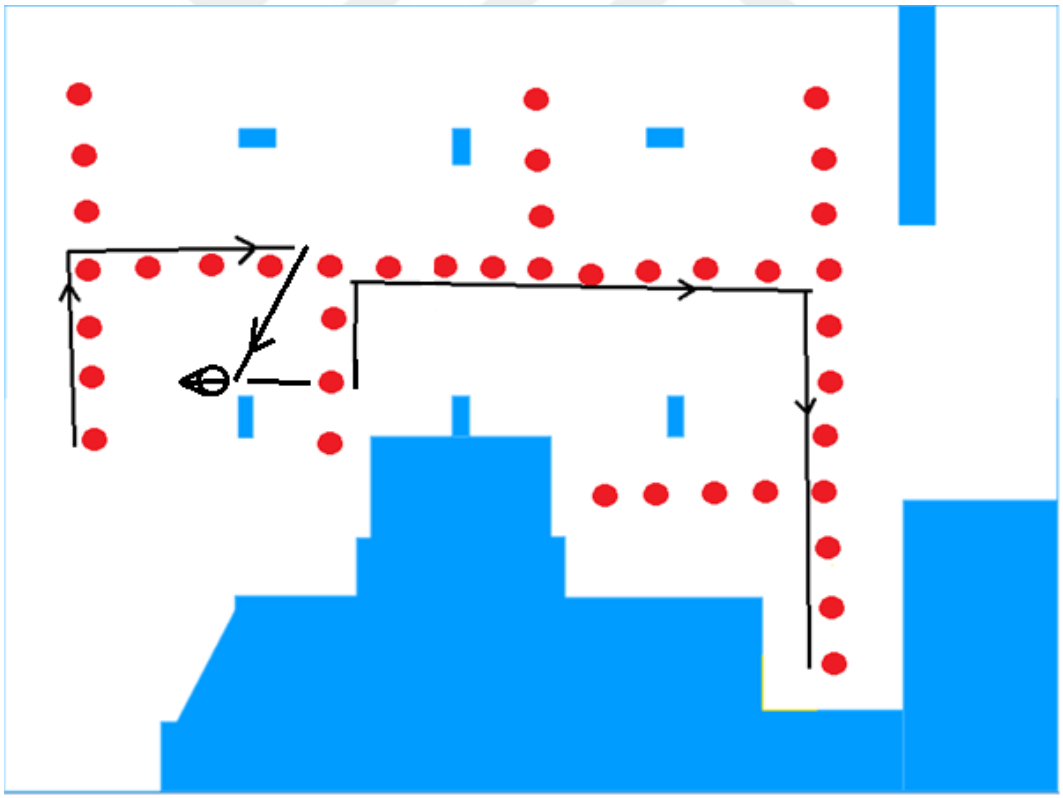
Görevdeki toplam hedef sayısı: 1

Görev toplam uzunluğu (metre): 25 (yoldan çıktığı yer hariç)

Görevdeki toplam dönme noktası sayısı: 2

28. Ara **yüzü kullanmada** ne derece başarılı oldu?
    a. Yardım almadan kullanabildi
    b. Bir kere yardım alarak kullanabildi
    c. İki – üç defa yardım alarak kullanabildi
    d. Üçten fazla yardım alarak kullanabildi
    e. Hiç kullanamadı
29. Ara yüzü kullanarak **amaç belirlemesi ne kadar sürdü** (saniye):……..

30. **Yönlendirme komutlarını ne derece doğru** uyguladı?
    a. Hiç şaşırmadan komuta göre hareket etti
    b. Bir kere şaşırdı
    c. İki – Üç defa şaşırdı
    d. Üçten fazla şaşırdı
    e. Komutların hiçbirini doğru uygulayamadı
31. Kaç kere **erişilebilirlik yolu dışına çıktı**:……
32. **Erişilebilirlik yolu dışına çıktığı durumların kaçında sistemi kullanarak tekrar yola girebildi**:……..
33. Kaç kere **erişilebilirlik yolu üzerinde rota dışına çıktı**:…….
34. Görevi tamamlama **hızı** (saniye):……
35. Hedeflerin **kaçına ulaşabildi**:…….
36. Kullanıcının **hareketleri**:



Kullanıcı Görüş ve Önerileri Anketi (test sonrası)

1. **Aşağıdaki maddelerde sistemin başarısı değişik kriterler bakımından sorgulanmıştır. Her bir maddede sorgulanan sistem başarı kriterini 1 den 5 e kadar derecelendiriniz. 5 çok iyi, 4 iyi, 3 orta, 2 kötü ve 1 çok kötü anlamına gelmektedir.**

1.1.Ara yüzün *amaç seçme* kolaylığı:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.2.Ara yüzdeki *butonlar* ile sistemi kullanabilme kolaylığı:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.3.Sistem siz amaç seçerken her işleminizden sonra bir sonraki adımda ne yapmanı gerektiğini bilgilendirmektedir. Örneğin, amacınızı seçtiğinizde eğer amacınızın bir koşulu varsa sistem size "amaç seçildi amaç koşulunu seçiniz" gibi sizden ne beklendiğini bilgilendirmektedir. Benzer şekilde diğer butonlar ile sistemi kullanırken de bilgilendirme yapmaktadır. Sistemin sizi amaç seçerken ve sistemi tanımlanmış butonlar ile kullanırken bilgilendirmesindeki başarısı:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)

e. 1 (çok kötü)

1.4.Erişilebilirlik yolunda bulunduğunuz sürece sistemin *BİP* sesi ile sizi bilgilendirmesindeki başarısı:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.5. Erişilebilirlik yolunda çıkmak üzereyken sistemin *titreşim* ile sizi uyarmasındaki başarısı:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.6.Sistemin, yanlış yöne gittiğinizde (erişilebilirlik yolundan çıkmadan) *hedef yolunu tekrar hesaplayıp* sizi yönlendirmesi:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.7.Sistemin, *erişilebilirlik yolundan çıktığınızda* sizi tekrar erişilebilirlik yoluna sokmak için yönlendirmesi:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)

e. 1 (çok kötü)

1.8. Sistemin, her hedef noktasından önce sizi *o hedef noktası ile ilgili olarak bilgilendirmesi:*

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.9. Sistem, ona girdi olarak verdiğiniz amacınıza uygun hedef noktaları belirlemekte ve bu hedef noktalarını da sizin en az yol yürümenizi ve önceden belirlenmiş bina veya yapacağınız işlem ile ilgili kurallara uymanızı sağlayacak şekilde sıralamaktadır. *Sistemin bu çalışma mantığının size göre kabul edilebilirliği:*

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

1.10. Sistemin, kullanıcıyı adım adım yönlendirmek için kullanıcıya "saat dokuz yönünde sola dön", "düz ilerle", "saat altı yönünde geriye dön" gibi komutlar vermesi:

a. 5 (çok iyi)
b. 4 (iyi)
c. 3 (orta)
d. 2 (kötü)
e. 1 (çok kötü)

2. **Sizce sistem akıllı yönlendirme gerektiren, aşağıda listelenmiş, karmaşık binalarda ne derece faydalı olur? Lütfen 1 den 5 e kadar derecelendiriniz.**

Hastanelerde:

a. 5 (çok faydalı olur)
b. 4 (faydalı olur)
c. 3 (az faydalı olur)
d. 2 (çok az faydalı olur)
e. 1 (hiç faydalı olmaz)

Üniversitelerde:

a. 5 (çok faydalı olur)
b. 4 (faydalı olur)
c. 3 (az faydalı olur)
d. 2 (çok az faydalı olur)
e. 1 (hiç faydalı olmaz)

Kütüphanelerde:

a. 5 (çok faydalı olur)
b. 4 (faydalı olur)
c. 3 (az faydalı olur)
d. 2 (çok az faydalı olur)
e. 1 (hiç faydalı olmaz)

Devlet binalarında (adliye, devlet kurumları…):

a. 5 (çok faydalı olur)
b. 4 (faydalı olur)
c. 3 (az faydalı olur)
d. 2 (çok az faydalı olur)
e. 1 (hiç faydalı olmaz)

Alışveriş Merkezlerinde:

a. 5 (çok faydalı olur)
b. 4 (faydalı olur)
c. 3 (az faydalı olur)

d. 2 (çok az faydalı olur)
e. 1 (hiç faydalı olmaz)

3. **Ek görüş ve önerileriniz:**