Travel Companion: A mobile system for trip assistance relying on Artificial Intelligence and Augmented Reality

Prodromos Lilitsis

Thesis submitted in partial fulfillment of the requirements for the Masters’ of Science degree in Computer Science

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes Campus, Heraklion, GR-70013, Greece

Thesis Advisor: Prof. Dimitris Plexousakis

---

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

This work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science.
University of Crete
Computer Science Department

Travel Companion: A mobile system for trip assistance relying on Artificial Intelligence and Augmented Reality

Thesis submitted by

Prodromos Lilitsis

in partial fulfillment of the requirements for the Masters’ of Science degree in Computer Science

THESIS APPROVAL
Abstract

The last years more and more people are traveling around the world. While car drivers are already assisted by advanced guidance and navigation facilities, continuous on-trip assistance for multimodal travelers is still in its infancy. Especially when it comes to situations of modal change, travelers get discouraged by the increased complexity and the lack of adequate information and guidance. Some of them face difficulties and feel stressed when they have to visit another place.

The goal of our research has been to integrate existing information systems, like Google Maps and Facebook, and to design and implement a prototype of a digital personal travel assistant for travelers. We recommend Travel Companion: A mobile system for trip assistance relying on Artificial Intelligence and Augmented Reality. Travel Companion assists a user through various guidance functions and offers personalized reminders and recommendations, that are based on user’s current location and information that is being collected from user’s mobile/tablet camera. It is mainly useful for people that feel insecure during their trips (e.g. Elderly, Disabled, people that do not know the local habits or the language of their destination).

Travel Companion selects routes for traveler’s trip that suits best to his personal characteristics or potential mobility problems. Traffic data is collected from Google Maps, while information that refers to personal preferences are collected from Facebook. Furthermore, personal information, like having a physical condition that limits their movements, are collected from an RDF Knowledge Base. We generate the best Travel Plan for user’s profile and provide directions to the user when he is located at an airport, underground station or bus station and provide notifications for important upcoming events or milestones in the trip. Our approach is based on Declarative Reasoning via a Formal language (Answer Set Programming), RDF and Semantic Web. These alerts could be activated either by scanning an airport sign (e.g. Check In) or by time limits.
Περίληψη

Τα τελευταία χρόνια ολοένα και περισσότεροι άνθρωποι ταξιδεύουν ανά τον κόσμο. Ενώ για οδηγούς αυτοκινήτων ήδη υπάρχουν συστήματα που τους βοηθούν να ταξιδέψουν και να πλοήγησουν με ασφάλεια, οι υπόλοιποι ταξιδιώτες δεν έχουν τόσες πολλές επιλογές για να βοηθήσουν. Ειδικότερα, όταν πρόκειται για κάποια σημαντική αλλαγή στο ταξίδι, οι ταξιδιώτες αποφασίζουν από την αυτοανόμενη πολυπλοκότητα και την έλλειψη πληροφοριών και καθοδήγησης. Μερικές φορές αγχώνονται όταν χρειάζονται να επισκεφθούν ένα άγνωστο μέρος.

Στόχος της έρευνας μας είναι να ενσωματώσει υπάρχουσεις πληροφορίες από άλλα συστήματα και να σχεδιάσει ένα πρωτότυπο ενός ψηφιακού ταξιδιωτικού βοηθού, ο οποίος θα είναι σε θέση να δίνει οδηγίες στο χρήστη υποκατάστατα στην πραγματική. Προτείνουμε τον Ταξιδιωτικό Βοηθό: Μια εφαρμογή για την εποπτεία ταξιδιώτες και βασιζόμενη σε Τεχνητή Νοημοσύνη και Επαυξημένη Πραγματικότητα. Το σύστημα υποστηρίζει τον ταξιδιώτη μέσω διάφορων λειτουργιών και προσφέρει προσωποποιημένες υπενθυμίσεις και καθοδήγησης, μεταξύ άλλων. Απευθύνεται κυρίως σε ανθρώπους που επιθυμούν να βρεθούν σε μια αναγνωρίσιμη και προσωπική θέση του ταξιδιού. Απευθύνεται κυρίως σε ανθρώπους που επιθυμούν να βρεθούν σε μια αναγνωρίσιμη και προσωπική θέση του ταξιδιού. Απευθύνεται κυρίως σε ανθρώπους που επιθυμούν να βρεθούν σε μια αναγνωρίσιμη και προσωπική θέση του ταξιδιού.
τεχνολογίες RDF και Σημασιολογικού Ιστού. Το σύστημα παρέχει ειδοποιήσεις στο
χρήστη, οι οποίες είναι σημαντικές ή χρονικά περιοριστικές. Οι ειδοποιήσεις αυτές
μπορούν να ενεργοποιηθούν είτε από τη σάρωση μιας ταμπέλας αεροδρομίου (Check
In), είτε λόγω χρονικών ορίων.
Acknowledgements

I wouldn’t have made it this far without the help and support of the kind people around me, to only some of whom it is possible to give particular mention here.

This thesis would not have been possible without the help, support and patience of my supervisors, Professor of the Computer Science Department Dimitris Plexousakis, Affiliated Researcher Giorgos Flouris and Postdoctoral Researcher Theodore Patkos of the Institute of Computer Science. They have been invaluable on both an academic and a personal level, for which I am extremely grateful. Our inspiring meetings and their excellent theoretical knowledge have let me profit a lot.

I would also wish to thank the Institute of Computer Science (ICS), Foundation of Research and Technology Hellas (Forth) and more specifically the members of the Information Systems laboratory (ISL) for the support, the goodwill to help and the great time we had during my graduate studies.

I particularly want to thank all my friends whose their care helped me overcome setbacks and stay focused on my study. I greatly value their friendship and I deeply appreciate their belief in me.

Last but not least, I would like to thank my parents Lefteris and Georgia for their personal support and great patience at all times. Furthermore, my brothers Vagelis, Manolis and Panagiotis who are there for me at any time, as always, for which my mere expression of thanks likewise does not suffice.
## Contents

1 Introduction .............................................. 3
   1.1 Motivation .............................................. 3
   1.2 Problem Statement .................................................. 4
   1.3 Contribution .................................................. 5
   1.4 Description .................................................. 7

2 Background .............................................. 9
   2.1 Answer Set Programming ........................................... 9
   2.2 Semantic Web ................................................ 9
   2.3 Ontology .................................................. 10
   2.4 Resource Description Framework .................................. 10
   2.5 Augmented Reality .............................................. 11
   2.6 Artificial Intelligence ........................................... 12

3 Related Work ............................................ 13
   3.1 Travel Assistants .............................................. 13
   3.2 Augmented Reality-Based Navigation Systems .................. 15
   3.3 Web Mapping Services ........................................... 16
   3.4 Indoor Navigation Solutions .................................... 17
   3.5 Comparison .............................................. 19

4 Methodology ............................................ 21
   4.1 Definitions .............................................. 21
   4.2 Components .............................................. 22
4.3 Architecture ......................................................... 23
  4.3.1 Trip Construction .............................................. 24
  4.3.2 Trip Monitor .................................................... 25
4.4 Ontology ............................................................ 26
  4.4.1 Users’ Profile ................................................... 28
  4.4.2 Trips and Tickets .............................................. 30
  4.4.3 Transportation Ways ......................................... 31
  4.4.4 Travel Segments ............................................... 34
  4.4.5 Places ............................................................ 37
  4.4.6 User’s Restrictions ............................................ 38
4.5 Reasoning Capabilities ........................................... 39
4.6 Augmented Reality ............................................... 40

5 Implementation ....................................................... 43
  5.1 Description ....................................................... 43
  5.2 User Profile ...................................................... 45
  5.3 Ticket Generator ................................................ 46
  5.4 Travel Segment Generator ..................................... 49
  5.5 Deadline Generator ............................................. 53
  5.6 Alerts ............................................................. 54
  5.7 Sign Detection using Augmented Reality ...................... 55

6 Use Cases ............................................................. 57
  6.1 A teenager travels from Heraklion to Athens ................. 57
  6.2 An elderly travels from Edinburgh to Melbourne ............. 59

7 Conclusions and Future Work ...................................... 67

A Documentation .......................................................... 73
  A.1 Environment ..................................................... 73
  A.2 Installation ...................................................... 74
    A.2.1 Server ....................................................... 74
    A.2.2 Client ....................................................... 74
A.3 Instructions ................................................. 75
A.4 Code Structure ............................................. 75
# List of Figures

1.1 Travel Plan from Heraklion to Dubai ........................................ 5

4.1 Knowledge Base ................................................................. 22
4.2 Scan Environment ................................................................. 22
4.3 Preferences ................................................................. 22
4.4 Transit ................................................................. 22
4.5 Reasoning ................................................................. 22
4.6 User Interface ................................................................. 22
4.7 Integration ................................................................. 22
4.8 Trip Construction Architecture ........................................... 25
4.9 Trip Monitor Architecture .................................................. 26
4.10 Trip Ontology ................................................................. 27
4.11 User Profile Ontology ....................................................... 30
4.12 Trips and Tickets Ontology ............................................... 31
4.13 Transportation Ways Ontology ............................................. 32
4.14 Travel Segments Ontology ............................................... 34
4.15 Places Ontology ................................................................. 38
4.16 User Restrictions Ontology .................................................. 38

5.1 Login Page ................................................................. 44
5.2 List of Trips ................................................................. 44
5.3 Travel Plan ................................................................. 44
5.4 Details of Travel Segment Stores ........................................ 45
5.5 Pop up Alert for time limit .................................................. 45
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Trip from Heraklion to Athens</td>
<td>58</td>
</tr>
<tr>
<td>6.2 Trip with John's tickets from Edinburgh to Melbourne</td>
<td>60</td>
</tr>
<tr>
<td>6.3 Trip with all tickets from Edinburgh to Melbourne</td>
<td>60</td>
</tr>
<tr>
<td>6.4 All Gates</td>
<td>64</td>
</tr>
<tr>
<td>6.5 Check In</td>
<td>64</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Popular Related Systems ........................................ 19  
3.2 Related Commercial Systems ................................. 20  

6.1 Available Tickets of Athens’s trip ........................... 57  
6.2 Available Tickets of Melbourne’s trip ...................... 60  

A.1 Development Versions ............................................. 73
Chapter 1

Introduction

1.1 Motivation

Nowadays with the proliferation of a panoply of smartphones and tablets on the market, almost everyone has access to mobile devices, which offers better processing capabilities and access to new information and services. The demand for new and more personalized services which aid users in their daily tasks is increasing. Current navigation systems lack in several ways in order to satisfy such demand, namely, accurate information about traffic or transit routes, possibility to personalize the information used by such systems and also the need to have a more dynamic and user friendly interaction between device and traveler. In short, the current navigation systems for travel assistance were not designed to be intelligent enough to communicate their needs effectively to humans, and they are not intelligent enough to understand travelers’ attempts to satisfy those needs.

Research on advanced traveler information systems show that travelers make better travel decisions, when they are well informed in a dynamic setting of urban public transport or airport system. However, the ability to be informed is not enough. People need to be able to quickly access and assess the information that is relevant to their own mobility. Unfortunately, most applications are not tailored or personalized to meet individual needs.

To the best of our knowledge there is no system that offers personalized travel guidance during multi-step journeys on public transport and airports – which will
include assistance with transfers from one place to another, and between different means of transport – and navigational guidance from the user’s stop on the public transport network, to his or her final destination. Our system collects information from user’s profile, Facebook Likes, and information from Google Maps API and provides traveler the best Travel Plan that fits to him, according to personalized information. A Travel Plan is a collection of actions that user should perform during his journey. This plan is generated according to rules that developed using Declarative Reasoning. In addition, our data are flexible and can be easily extended, due to our method. Our approach has been carefully designed in order to be easily flexible and extensible to assist either more user categories (visually impaired, children) or more places (ports). For instance, an Elderly is able to climb some stairs, whereas a person with mobility problems is not. Such restrictions can be easily described, using our approach, in order to provide assistance.

Travel Companion controls the whole trip and will provide the user with detailed directions, like “Now you should find your bus ticket”. Our approach focuses on helping the user to reach his destination using information from the tickets, and collecting information from pictures and signs that may be acquired using augmented reality. Travel Companion offers personalized services by including data from Facebook (likes) and Google Maps (routes) and offers real-time alerts. Data from different domains are computed based on explicit rules and how mobile/tablet can dynamically adapt to travelers’ surrounding environment and activities in order to maximize the relevance of the displayed information.

1.2 Problem Statement

Usually, people with lack of travel experience face problems during a trip. Elderly, Disabled or any other users that may feel insecure during a trip may need a Travel Companion for assistance.

For instance, when a Greek travels from Athens to Dubai may does not speak Arabic, so our traveler could not follow Arabic instructions or signs in order to navigate in this airport. Furthermore, when someone travels from Athens to California he usually owns one ore more air tickets. However, he is going to need more
1.3. CONTRIBUTION

transportation means in order to get in California. Such tickets need to be found during their trip and travelers need assistance in order to find them. Furthermore, people usually get lost in airports because they do not have travel experience and do not know how to find their gate, or even what should they visit first "Check In" or "Luggage Control". They need a system to assist them in finding such routes and provide a plan for their trip.

There is lack of systems that provide assistance during a long journey. For instance, a system that offers assistance to travelers that travel from Heraklion to Dubai would be useful. Users need someone to assist them with a plan like in figure 1.1 and using a smart-phone’s camera they can acquire assistance from signs that cannot read, because Arabic is an unknown language for them. Last but not least, a system that provides personalized assistance for elderly or disabled would be a relief for them.

1.3 Contribution

Since there is no other similar approach to assist different user categories and provide personalized Travel Plans, our approach extends the proposed systems and offers more features.

First of all, personalized deadlines are assigned to each portion of the trip.
Specifically, each user, depending on age or mobility, may need more or less time in order to perform his obligations in an airport or station. For instance, elderly may need more time during Check In and less time during walking routes (walking to Gate in an airport). Furthermore, when travelers carry a luggage may need even more additional time during walking routes.

Personalized Travel Segments\textsuperscript{1} are generated, for each user category, in order to provide assistance for both indoor and outdoor place. For instance, if a user carries a luggage, Travel Companion should remind him to get at "Luggage Claim" area after getting off the plane. Travel Segments adapt to places too. For example, if traveler is located in an airport Travel Segments should be different from a subway’s station.

System provides personal recommendations for spare time, during waiting time, in Duty Free or stores near stations, based on Facebook’s preferences. For instance, if a traveler is located near Starbucks and we have information from Facebook that he likes Starbucks, we offer it as an option during waiting time. In addition, Travel Companion offers directions that provide user the best option to reach a place. For example, such personalized directions could be how to reach a platform using either stairs or elevator (depending his mobility).

Travel Companion prefers using faster Transportation means for routes, in order to avoid user from getting tired. We choose routes with fewer transfers when our traveler is an elderly or disabled and keep track of time in order to notify the traveler in case of a deadline. Finally, using Augmented Reality user is able to see more about his location. For instance, detecting a sign in an airport, it may activates an alert that provides helpful directions for his route.

More specifically, our main contributions can be summarized as follows:

- Adaptable deadlines to each user category
- Travel Segments adapted to user category
- Travel Segments adapted to user’s stuff, additional Travel Segments are generated if a luggage is carried

\textsuperscript{1}Travel Segments refer to a portion of a trip, like a Bus Stop or Check In.
1.4. **DESCRIPTION**

- Travel Segments adapted to the place (airport, subway station, etc)
- Personal recommendations for spare time in Duty Free or Stores
- Alerts adapted to user’s category when needed
- Best Public Transportation routes that suits to user’s category
- Time sensitive Alerts activated when needed
- Assistance via Augmented Reality, providing Alerts

Last but not least, our architecture is adaptable and extendable for more user categories or places. As, we have designed our approach using an RDF Knowledge Base and Declarative Reasoning, as described in sections 5.2, 5.3, 5.4, 5.5 and 5.6, Travel Companion can be easily adapt either for other users, like children, visually impaired people or possible modification of Travel Segments.

1.4 Description

We below describe the Background of this thesis and any information or technology that is essential to solving this problem. Afterwards, we describe Related Work and similar approaches of Travel Companions that offer assistance during trips. Then, we propose our Methodology and our Implementation that finally developed in order to provide the best solution of a Travel Companion to this problem. Then, we describe some Use Cases, that clarify the use of our system and Conclusion and Future Work. Finally, an Appendix about our implementation’s documentation.
Chapter 2

Background

2.1 Answer Set Programming

*Answer set programming* (ASP) is a form of declarative programming oriented towards difficult search problems [1]. It is based on the stable model "answer set" semantics of logic programming. In ASP, search problems are reduced to computing stable models, and answer set program solvers for generating stable models are used to perform search. The computational process employed in the design of many answer set solvers is an enhancement of another algorithm (Davis – Putnam – Logemann – Loveland [2]) and, in principle, it always terminates. ASP offers Clingo, Gringo and Clasp. We used Clingo, a reasoner, because it combines both gringo and clasp into a monolithic system, offering more control over the grounding and solving processes.

2.2 Semantic Web

*Semantic Web* is an extension of the current Web that provides an easier way to find, share, reuse and combine information [3]. It is based on machine-readable information and builds on XML technology’s capability to define customized tagging schemes and RDF’s (Resource Description Framework) flexible approach to representing data. The Semantic Web provides common formats for the interchange of data (where on the Web there is only an interchange of documents). It also
provides a common language for recording how data relates to real world objects, allowing a person or a machine to start off in one database, and then move through an unending set of databases which are connected not by wires but by being about the same thing.

2.3 Ontology

Ontology is a formal naming and definition of the types, properties, and interrelationships of the entities that exist for a particular domain of discourse [4]. In order to implement such a representation, several languages have been developed. The one that currently gets the most attention is probably the Web Ontology Language (OWL). OWL is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things and relations between things.

In a traditional relational database, concepts can be stored using tables, but the system does not contain any information about what the concepts mean and how they relate to each other. Ontologies do provide the means to store such information, which allows for a much richer way to store information. This also means that one can construct fairly advanced and intelligent queries. Query languages such as SPARQL have been developed specifically for this purpose.

2.4 Resource Description Framework

Resource Description Framework (RDF) is a general framework for how to describe any Internet resource such as a Web site and its content [5], [6]. An RDF description (such descriptions are often referred to as metadata, or "data about data") can include the authors of the resource, date of creation or updating, the organization of the pages on a site (the sitemap), information that describes content in terms of audience or content rating, key words for search engine data collection, subject categories, and so forth. The Resource Description Framework makes it possible for everyone to share web-site and other descriptions more easily and for software developers to build products that can use the metadata to provide better search engines and directories, to act as intelligent agents, and to give Web users more
2.5. AUGMENTED REALITY

control of what they're viewing. The RDF is an application of another technology, the Extensible Markup Language (XML), and is being developed under the auspices of the World Wide Consortium (W3C).

An RDF store is a purpose-built database for the storage and retrieval of triples through semantic queries. A triple is a data entity composed of subject-predicate-object, like "John knows Prodromos", where "John" is the subject, "knows" is the predicate and "Prodromos" is the object.

Much like a relational database, one stores information in a triplestore and retrieves it via a query language. Triplestore is a type of graph database that stores semantic facts (triples). A triple contains three components:

- The subject, which is an RDF URI reference or a blank node
- The predicate, which is an RDF URI reference
- The object, which is an RDF URI reference, a literal or a blank node

Unlike a relational database, a triplestore is optimized for the storage and retrieval of triples. In addition to queries, triples can usually be imported/exported using Resource Description Framework (RDF) and other formats.

2.5 Augmented Reality

Augmented Reality is the integration of digital information with the user’s environment in real time [7]. AR uses the existing environment and overlays new information on top of it and allows developers to tie animation or contextual digital information in the computer program to an AR "marker" in the real world. AR applications for smartphones/tablets typically include global positioning system (GPS) to pinpoint the user’s location and compass to detect orientation.

Image recognition is the process of identifying and detecting an object or a feature in a digital image or video. This concept is used in many applications like systems for factory automation, toll both monitoring, and security surveillance. For instance, when an airport sign is detected user should be able to see more information about it using his smart-phone.
2.6 Artificial Intelligence

Artificial Intelligence (AI) is the area of computer science that emphasizes the creation of intelligent machines that work and react like humans. AI has actually become an essential part of the technology industry. In computer science, the field of AI research defines itself as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of success at some goal. Colloquially, the term "Artificial Intelligence" is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as learning and problem solving.
Chapter 3

Related Work

There is a plethora of tools and applications that provide directions and accompany people when they need to reach an unfamiliar place. Since our approach is based on maps and travel companions applications, below we first discuss the most relevant travel assistant systems, we then report some web mapping services, after that some Indoor Navigation Solutions that provide people directions in a building, and finally we propose some recommendation systems.

3.1 Travel Assistants

Travel assistants advise travelers on their destinations and make recommendations for transportation, or accommodations. For travelers who are going abroad, agents will provide information on custom regulations.

Samuli Heinonen and Erkki Siira [8] propose the ASSISTANT for elderly that empowers users to use public transportation in a way to maintain activity and mobility. The system allows users to pre-book a taxi, a public transportation or a staff assistance in transportation hubs.

Similar to ASSISTANT, [9] offers real time information about the route. Such information are estimated time to the next stop of the route or estimated time to the traveler destination, also if desired, tourist information about nearby interesting points. The application is designed for all kind of passengers, but it is especially useful for travelers with special needs or for those who are unfamiliar with a public
transport network. However, the user needs to spend money in order to cope with problems or difficulties in transportation hubs and does not provide any personalized routes. For instance, disabled can easier travel with subway rather than bus.

[10] proposes PASSAGE travel assistant that integrates Atlanta-based crime data to find "safe paths" between any given start and end locations in Atlanta. It also provides security features in a convenient user interface to further enhance safety while walking. However, PASSAGE does not offer any guidance for airports or public transportation.

Viamigo [11] is a digital travel assistant that monitors people with intellectual activities on the move, evaluates their position against the expected path and alerts caretakers or coaches when unexpected events occur. [12] offers step-by-step assistance along predefined routes for intellectual people within a city. It can assist not only with learning new travels but also to complete tasks such as house chores or work tasks. However these routes should be short and need to be generated by caretakers in order to provide the suitable guidance.

H. Pan, C. Yi and Y. Tian propose [13] a traveling assistant system of bus detection and recognition for visually impaired people. It is able to notify the visually impaired people in speech the information of the coming bus, and detect the route number and other related information. However, it does not provide any information about other public transportation, and it is not able to generate routes or Travel Plans for long trips.

[14] proposes a personal mobility assistant that promotes use of public transport by helping user identify the best travel option across a multi-modal transport network, through a user-friendly interface that intelligently adjusts to user preferences and to contextual information. This approach aims to reduce energy consumption, lower emissions and pollution levels and improve public safety. The user can choose and filter routes depending the price or traffic. However, this approach does not offer routes and plans for airports and does not provide special route for people with mobility problems or elderly.

[15] and [16] provide location aware travel assistance for city tourists based on a
3.2. **Augmented Reality-Based Navigation Systems**

A combination of widespread conventional mobile phones with built-in cameras and GPS identifies sights by taking photos, leads tourists to sights matching their interests and lets them collect multimedia impressions with their mobile phone along the tour. However, these applications do not provide any personalized Travel Plans. For instance, people with mobility problems require different kind of guidance.

Our system Travel Companion extends the above functionalities and offers an easily extensible approach for long-distance trips that is able to offer personalized Travel Plans and routes.

**3.2 Augmented Reality-Based Navigation Systems**

*Insider* [17], *Wikitude* [18], *Image-based Positioning System* [19], *Real View Navigation* [20] and *AR Navigation* [21] offer augmented reality-based navigation without the use of special hardware. Using Augmented Reality and navigation technologies worldwide unique holistic solution which enables clients to easily make their place recognizable for mobile devices and gain added value with Augmented Reality. However, these approaches do not provide any personalized routes. For instance, disabled people require different kind of guidance (need to use an elevator instead of stairs).

[22] and [23] offer indoor navigation using Augmented Reality and interior features to determine the user’s location and provide navigation instructions. They propose a smartphone or wearable technologies, like Google Glasses, as navigation devices for better evaluation results. However, these approaches have not been extended for outdoor navigation.

[24] investigated the use of Mobile Augmented Reality technologies to support indoor navigation for wheelchair individuals. This approach offers routes that allow users to avoid obstacles, like stairs or narrow paths. These points need to be marked in order to be avoided by such users. However, this approach works for buildings that have been equipped with specific markers and does not offer any guidance for outdoor navigation.

[25] demonstrates markers are superior to other methods in terms of positioning.
accuracy and cost. Using Augmented Reality Markers that has been placed in buildings, users can easily track these markers and navigate in a building. These markers can be visible with a smart phone’s camera, so the system can easily detect user’s location. Nevertheless, this approach does not offer any guidance for users with mobility problems or offer directions to outdoor environments.

However, these approaches have not been extended for both indoor and outdoor navigation. Our approach is able to provide personalized navigation instructions for a traveler, wherever he is, and prevents him of following wrong directions.

3.3 Web Mapping Services

Web mapping is the process of using maps delivered by Geographic Information Systems (GIS). A web map on the World Wide Web is both served and consumed, thus web mapping is more than just web cartography, it is a service by which users may choose what the map will show. Web GIS emphasizes geodata processing aspects more involved with design aspects such as data acquisition and server software architecture such as data storage and algorithms, than it does the end-user reports themselves. Web mapping has brought many geographical data-sets.

[26], [27], [28], [29], [30], [31], [32] and [33] offer satellite imagery, street maps, panoramic views of streets, real-time traffic conditions and route planning for traveling by foot, car, bicycle, or public transportation. These Web Mapping Services combine multiple means of transport except for transportation among airports. They can provide detailed directions to users that will help them to reach their destination.

However, these navigation tools focus on mapping the available routes and it is the user that should decide the best choice for him. These take into account information about user requirements in order to provide a more friendly way of navigation. For example, if there is a disabled passenger, he should have been assisted with different directions from an elderly one.

All of them focus on traffic and shortest paths, using public transportation, walking or driving, ignoring user’s requirements. For example, some routes may not be achievable for elderly or disabled. Our approach collects and filters multiple
routes in order to provide the best one that can be successfully performed for our traveler.

3.4 Indoor Navigation Solutions

An indoor positioning system (IPS) is a system to locate people inside a building using radio waves, NFC, acoustic signals, or other sensory information collected by mobile devices. GPS reception is normally nonexistent inside buildings, so other positioning technologies are used here when automatic positioning is desired. Other sensory information are often used in this case to create a so-called "indoor GPS". Contrary to GPS, however, they also enable you to determine the actual floor level. Most applications require an "indoor routing" functionality that guides people precisely through a building using an indoor navigation app and in this way, automatically determines their position - very similar to the navigation systems that we use in our cars. A typical application is turn-by-turn navigation in an app (displaying directions on a digital map) used for train stations, airports, shopping centers and museums. This kind of application can also include many other useful services.

[34] is a map platform for organizations that located in large buildings. This system offers indoor maps, information about meetings or events that take place in specific rooms, and directions to reach a place, using a mobile device. [35], offers indoor directions for large buildings, that have been created and installed, using Google Maps [26]. Another research proposes preinstalled points of interest on an indoor map and using a mobile device with acceleration sensors in real time the system can detect users location and navigate him [36].

However, these approaches need to have predefined and mapped any place in order to navigate successfully and does not take into account any personal information or needs. For instance, some users may not perform a long distance walk, so the system should provide alternative routes.

[37], [38] provide personalized indoor navigation for visually impaired people, using communication technologies like WiFi and radio-frequency identification, in order to locate and identify users in the building. [39] and [40] offer a intelligent
wheelchair equipped with sensors that can relocate users into a crowded place avoiding obstacles. However, these approaches do not provide any navigation plan and it is up to user to decide the place of destination. For instance if a user is located in an airport and travels to an unfamiliar place, the system will not provide further directions about Check In or Gate.

ppNav [41] offers navigation services avoiding the requirements of pre-deployed location services and detailed floorplans. It actually navigates a user to the destination by tracking user mobility, promoting timely walking tips and alerting potential deviations, according to a previous traveler’s trace experience. However, this approach performs effectively for building that have been visited by ppNav’s users.

These approaches focus on mapping large buildings, or mark specific points in order to provide navigation instructions. Our approach extends the proposed systems and provides personalized and easily extensive navigation plans for any user for both indoor and outdoor assistance.
3.5 Comparison

The table below summarizes the most relevant approaches:

<table>
<thead>
<tr>
<th>Systems</th>
<th>User Categories</th>
<th>Functionality</th>
<th>Augmented Reality</th>
</tr>
</thead>
<tbody>
<tr>
<td>[8] Elderly</td>
<td>Public</td>
<td>Google Maps</td>
<td>No</td>
</tr>
<tr>
<td>[13] Visually Impaired</td>
<td>Bus</td>
<td>Bus Detection</td>
<td>No</td>
</tr>
<tr>
<td>[14] -</td>
<td>Public</td>
<td>Green Navigation</td>
<td>No</td>
</tr>
<tr>
<td>[37] Visually Impaired</td>
<td>Indoor</td>
<td>Wifi/Sensors</td>
<td>No</td>
</tr>
<tr>
<td>[38] Blind</td>
<td>Indoor</td>
<td>Wifi/Sensors</td>
<td>No</td>
</tr>
<tr>
<td>[39] Disabled - Elderly</td>
<td>Indoor</td>
<td>Wheelchair</td>
<td>No</td>
</tr>
<tr>
<td>[41] -</td>
<td>Indoor</td>
<td>Machine Learning</td>
<td>No</td>
</tr>
<tr>
<td>[25] -</td>
<td>Indoor</td>
<td>AR Markers</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 3.1: Popular Related Systems

These approaches does not offer directions for both indoor and outdoor places, but only focus to one of them. Furthermore they do not provide routes for large distance trips, they focus on assisting in a building or in a city. In addition, none of them provide assistance in airports or using Augmented Reality, in order to direct user in the right place. Finally, these approaches are not flexible and extensible, in order to assist more user categories or places, like airports, because they have not used any Semantic Web technology.

Most relevant related systems that found in Play Store and Apple Store are summarized below:
CHAPTER 3. RELATED WORK

These systems provide directions either for indoor or for outdoor environments and can be used as they are. They do not provide any information about their background, in order to conclude about their expansibility.
Chapter 4

Methodology

4.1 Definitions

In this dissertation we describe our system’s operation and make use of some terms, which are described below:

- **Trip**: is an act of traveling from one place to another. Each trip includes one or more tickets or walking routes, using transportation means and/or airports.

- **Ticket**: is a card that gives the holder a certain right, especially to travel by public transport or airplane.

- **Transportation Means**: any of bus, train, subway, airplane, or taxi.

- **Public Transportation**: any of bus, train, metro.

- **Travel Segment**: is a term that refers to a portion of a trip. For instance, "Check In" is a Travel Segment of an air ticket. Each ticket consists of one or more Travel Segments.

- **Travel Plan**: is the aggregation of trip’s Travel Segments.

- **Place**: is any area that relates to a trip. It can be either indoor or outdoor area, like a "Bus Stop" or "Check In" in an airport.
4.2 Components

Travel Companion was built using seven different components. These parts are in collaboration for providing a Travel Plan for our traveler. These components are shown in Figures below:

- **Figure 4.1:** Knowledge Base
- **Figure 4.2:** Scan Environment
- **Figure 4.3:** Preferences
- **Figure 4.4:** Transit
- **Figure 4.5:** Reasoning
- **Figure 4.6:** User Interface
- **Figure 4.7:** Integration

Figure 4.1 describes an RDF knowledge base that stores any information, using Blazegraph. Such information are user’s profile, trips, tickets and any other information relates to user. Any data are requested using SPARQL queries from 4.7 component and a Json response is sent back.

Figure 4.2 offers the feature to detect images with a mobile’s phone camera, using Vuforia API. In case of requesting assistance via Augmented Reality user is about to use this feature. Vuforia is receiving requests from 4.6, where frames from camera are collected and sent to Vuforia for recognition.

Figure 4.3 relates to Facebook likes. In order to provide recommendations
during waiting time at airports or stations we offer specific shops that match to user’s profile. During a user registration 4.7 component requests user preferences about cafes and restaurants. After that a Json response is received and the response is stored back to 4.5 component, where there is a file for each user that holds predicates for user’s profile.

Figure 4.4 is based on Google Maps. In order to load any available Public Transportation for routes that actually our traveler does not own any ticket, we provide them from Google Maps (if available). 4.7 component requests its execution in order to load such routes. A Json response is then parsed and these data are store in 4.1.

Figure 4.5 is the core of our system. Using Declarative Reasoning we provide an easy and extensible way to generate Travel Plans, depending to many different domains. Component 4.7 requests its execution and the generated plan is then parsed from 4.7 component and stored back to 4.1.

Figure 4.6 is a User Interface that developed using Unity. It offers an easy way to interact and keep track of the trip. It requests and exchange information in Json format with 4.7 component.

Figure 4.7 is the Java part that is responsible for integration and connection among components. It sends SPARQL queries to 4.1 and receives a response in Json format. It exchanges any information with the rest components in Json Format and communication with component 4.6 is performed using Restful Web Services and Web Sockets.

4.3 Architecture

Our aim is to provide an adaptable and personalized plan for our traveler. Depending his age or his mobility problems using Declarative Reasoning, system will provide the best Travel Plan. As presented in section 4.2 system is composed by seven modules, that we are going to describe how do they work together.

In this section, we present the architecture of our system. Figures 4.8 and 4.9 describes the architecture of the Travel Companion.
4.3.1 Trip Construction

First of all, as shown in figure 4.8 our traveler registers in the application using credentials from a social media account. Then, the user can login into the application using the social media credentials, from the previously registered social media account. Afterwards, he can add any ongoing trip by scanning all the available tickets. The necessary information of the user’s profile and his trips are recorded on a Knowledge Base. Any scanned tickets will provide the necessary information about the overall trip of the user. The application, as a next step, will calculate any missing and necessary routes in-between the routes that belong to the scanned tickets. As an example, we can imagine that a user can scan two tickets, one that belongs to a flight from Heraklion to Athens and a second one that belongs to a flight from Thessaloniki to London. The route gap between Athens and Thessaloniki is a set of routes that the application will provide, using Google Maps.

The Knowledge Base and a maps API will provide all the available travel routes which will potentially fill the gaps for the user’s routes. Since the maps API provides any available routes that correspond to our user’s in-between travels, Java (4.7) language will sort and filter these routes with regards to the user’s profile which is stored on the Knowledge Base. Rules that correspond to the user’s profile and his data will be applied and eventually the application will suggest the final travel routes which the user can follow for his in-between travels to his final destination. Then, the user should proceed by buying the remaining tickets for the new proposed travel routes. This will now constitute a new set of tickets, that includes the initial scanned by the user and those that the user need to buy for the proposed travel routes. Everything is recorded back to the knowledge base.

Given the fact that a travel route might demand different actions by the user, Travel Segments are generated by the application. These Travel Segments are operations that describe the user’s actions during a travel route. All the Travel Segments will be generated in combination with the user’s profile and data (which are provided by the Knowledge Base) in accordance with rules using Declarative Reasoning.
4.3. ARCHITECTURE

4.3.2 Trip Monitor

During the trip user follows the Travel Plan and system provides assistance using either Augmented Reality or Alerts. Last but not least, ASP will combine the Travel Segments with any potential changes on the Knowledge Base.

For example, let us assume that a Travel Segment guides our user to an airport gate. At some point the user does not know its current location in the inner space of the terminal. So, through the application there is the ability for the user to use the smart-phone camera and scan the surrounding environment. The application camera feature will provide an Augmented Reality interface, where the user is able to see and understand his next move in order the Travel Segment to get completed. For this example the camera will point the user to the right direction of the airport.
gate with an alert. The system will generate this alerts while, simultaneously, the
Knowledge Base will be updated with the states of the running Travel Segment.

Furthermore, when user is located in Duty Free or shops, through the Augmented Reality interface is able to accept recommendations using his social media preferences. System filters information that is being collected from camera and offers to user the best option during waiting time at airports or stations.

![Trip Monitor Architecture](image)

Figure 4.9: Trip Monitor Architecture

### 4.4 Ontology

In order to limit the complexity of our data and organize our stored information, we developed an ontology. We propose Trip Ontology that is the formal naming and definition of our data.

Our Trip Ontology, developed using RDF, offer the features below:

- Sharing common understanding of the structure of information among people or software agents
- Easily enable reuse of domain knowledge
- Make domain assumptions explicit
- Separate domain knowledge from the operational knowledge
4.4. ONTOLOGY

- Analyze domain knowledge

Figure 4.10 is the whole Trip Ontology that we propose. The next sections describe in detail the form of our RDF data.

Figure 4.10: Trip Ontology
4.4.1 Users’ Profile

Our approach offers user dependent directions and preferences. Such information are described from the figure 4.11. Our approach offers assistance for elderly, disabled and common users and could easily offer assistance to more by adding more subclasses to figure 4.11. Each user carries one or more luggage depending its weight and size. We should mention that our approach respects any user category and a common user is not considered more skillful than the rest. These categories are:

- **Elderly**: any person above 68 years old
- **Disabled**: any person with physical condition that limits his movements
- **Common**: any person that does not actually have any physical or mental disability

For each category we have also defined some properties which are described below:

- **Max Walking Distance**: the maximum distance each user category is able to walk
- **User Extra time**: the additional time that need to be added for each user’s Travel Segment

For instance, an Elderly could not walk more than 400 meters so our system should know that in order to find him another way to reach his destination (Transportation Mean). Furthermore, each category may need further time in order to perform an action, like pass through the luggage inspection area, so Extra User time property represents that time in minutes.

In addition, each user often carries luggage, as shown in figure 4.11. We propose two types of luggage:

- **Hand**: is any luggage that actually fits in a plane cabin
- **Heavy**: is any luggage that does not fit in a plane cabin
4.4. ONTOLOGY

It is useful to have such information for the user in order to provide the best personalized route according to his profile. For instance, if he carries a Heavy luggage we should direct him to check in first in an airport environment.

Each luggage consists of two properties:

- **Weight**: is the weight of a luggage in kilos
- **Luggage Extra Time**: is actually the additional time that a user needs in order to perform an action when carries a luggage

If a luggage is more than nine kilos, then is considered as "Heavy", so we should generate further actions for the traveler and, of course, carrying a luggage often causes delays to users, that need to be taken into account.

Last but not least, **User** is any registered user that actually carries luggage and belongs to one of the above categories and includes four more properties:

- **Date of Birth**: is the birth date
- **Email**: email address
- **Name**: name and surname
- **Password**: his personal password

Such information are used to conclude traveler’s age and preferences. We use email and password in order to load social media personal preferences in order to provide the best recommendations when needed.
4.4.2 Trips and Tickets

A trip consists of one or more tickets that each of them relates to a source and a destination place as shown in figure 4.12. A traveler can owns to one or more trips and each trip consists of one ore more tickets. For each trip we have defines the properties below:

- **Starting Date**: is the date and time that the trip is about to begin
- **Expiration Date**: is the date and time that the trip is about to be completed
- **Is Active**: is the state of the trip. It can be active or inactive
- **Source Name Place**: is the address of source place
- **Destination Name Place**: is the address of destination place

A ticket includes information that will allow us to handle them using time, user profile and the related Transportation Mean. Below are some properties that provide such information:

- **Departs At**: is the date and time of departure
- **Arrives At**: is the date and time of arrival
4.4. ONTOLOGY

- **Is Available**: defines if the traveler holds the ticket or not (should buy it)

- **Departs From**: is the address of source place

- **Arrives To**: is the address of destination place

Using these information we are able to know not only user’s routes but also tickets and places that will be visited.

![Figure 4.12: Trips and Tickets Ontology](image)

**4.4.3 Transportation Ways**

We designed an ontology in order to describe the different Transportation Ways that a user can use in order to perform an action. As shown in figure 4.13 we propose two different categories of Transportation Ways:

- **Transportation Means**: is any outdoor transportation mean

- **Atomic Transportation**: is any way of indoor transportation

These categories describe transportation means for indoor and outdoor places. Indoor navigation at airports or stations and outdoor using Public Transportation or taxi.
Firstly, *Transportation Means* consists of five sub-classes, which are described below:

- **Airplane**: relates to air tickets
- **Bus**: relates to bus tickets
- **Metro**: relates to underground tickets
- **Train**: relates to train tickets
- **Taxi**: relates to routes that can not be performed with the transportation above

Figure 4.13: Transportation Ways Ontology
4.4. ONTOLOGY

In order to distinguish tickets for different Transportation Means we link a ticket with one of the above sub-classes, including a property *Time Before Departure*, that defines how many minutes earlier a traveler should be there before departure, and *Has Preference Order*, that defines the order of the public transportation that we are going to prefer when there are more than one available.

Secondly, *Atomic Transportation* consists of seven sub-classes, which are described below:

- *Wheelchair*
- *On Foot*
- *Escalator*
- *Elevator*
- *Short Stairs*
- *Long Stairs*
- *Airport Car*

*Short Stairs* can be found in a bus, while *Long Stairs* could be in an Underground Station, so we would avoid using this option for an Elderly. Big airports provide an electric car for people that are not able to walk and delivers them to their terminal.

These *Atomic Transportation* can be found in an airport, or any station. Some user categories are able to use one or more of these Transportation Means. For instance, *Elevator* can be used be anyone, but an *Airport Car* should be used from people with limited notabilities. *On Foot* subclass includes a property *Extra Time* that defines, in minutes, additional time that a user need to perform an action on foot. Likewise, for *Wheelchair* subclass there is a property *Extra Time* that defines, in minutes, additional time that a user need to perform an action using a wheelchair.
4.4.4 Travel Segments

*Travel Segment* is an action that our traveler need to complete in order to reach his destination. For example, a bus ticket consists of some actions. The first one is "Kiosk", which recommends user to buy a ticket. An action could be visiting some stores in Duty Free, if there is enough time, and then get to your "Gate". Such actions are generated according to the ticket type and user’s category and profile. These actions are described in figure 4.14. We will refer as Travel Plan, a collection of Travel Segments.

Figure 4.14: Travel Segments Ontology

We propose the sub-classes below of *Travel Segments*.

- **Check In**: is the process whereby people announce their arrival at an airport
- **Duty Free**: shops are retail outlets that are exempt from the payment of certain taxes and can be found at airports
• **Claim Hand Luggage**: is the process whereby passengers need to claim his cabin luggage from aircraft’s compartments

• **Claim Heavy Luggage**: is the process whereby passengers need to claim his heavy luggage from airport’s reclaim area

• **Drop Hand Luggage**: is the process whereby passengers need to place his cabin luggage in aircraft’s compartments

• **Drop Heavy Luggage**: is the process whereby passengers need to provide their luggage at airline’s check in desk

• **Contact Airport Service**: is the process whereby passengers call for help in the airport

• **Contact Flight Attendant**: is the process whereby passengers call a flight attendant for help in the aircraft

• **Get to Gate**: is the process whereby passengers need to get to their Gate

• **Airport Exit**: is the process whereby travelers need to reach airport’s exit

• **Aircraft Exit**: is the process whereby travelers need to reach aircraft’s exit

• **Luggage Inspection**: is the process whereby travelers will be security screened

• **Take your Seat**: is the process whereby a passenger needs to find his seat in an airplane

• **Go to Kiosk**: is the process whereby a traveler need to go and buy a ticket for a public transportation

• **Stores**: is the process whereby a traveler may spent some time at stores before departure

• **Start Line**: is the process whereby a traveler needs to find his departure point (Platform, etc)
• **Destination Line**: is the process whereby a traveler needs to get off his current public transportation in the next stop, Platform, etc.

All these categories of a **Travel Segment** are different actions that a user may need to perform, based on his profile or ticket, in order to reach his destination.

Furthermore, each **Travel Segment** is characterized by some additional properties like:

• **Type Name**: is a 2-3 worlds description of the action

• **Has Duration**: is the duration in minutes of each different type of segment

• **Deadline**: is the latest time and date by which the action should be completed

• **Is Optional**: defines whether the action is optional or not

• **Has Order**: is the number that defines the execution order of each action

• **Has ASP Name**: is the predicate that links this segment with Answer Set Programming. Travel Segments are generated using ASP and each Travel Segments is assigned to a rule. This property is the rule, in order to easily translate rule in Travel Segments.

All **Travel Segments** are performed with specific order, so a user could perform only one segment every moment, without breaking the order. So, each **Travel Segment** Has a specific state. Such states are:

• **Canceled**: declares that the action has been canceled

• **Pending**: declares that the action is pending

• **Executing**: declares that the action is currently performed

• **Completed**: declares that the action has been successfully executed and completed

Finally, each state is assigned to an Answer Set Programming predicate, which is the property **Has ASP State Name**. Of course, each **Travel Segment** relates to a specific place, that we are going to propose in the next subsection.
4.4.5 Places

A *Place* is a particular position, point or area. We have defined six different types of places. As shown in figure 4.15 such types are:

- *Taxi Point*: is any place that relates to a cabstand
- *Bus Point*: is any place that relates to bus, like a bus stop or bus station
- *Airport*: is an airway with extended facilities for the take-off, landing, and maintenance of civil aircraft, with facilities for passengers
- *Metro Point*: any railway station for a rapid transit system where passengers can get on and off metros
- *Train Point*: any place where passengers can get on and off trains
- *Service Point*: any place that a user can rest or spend some time, like a hotel, department stores, duty free

Furthermore, there are additional properties that have been defined for each *Place*. Such properties are:

- *Longitude*: is the angular distance of a place east or west of the Greenwich meridian
- *Latitude*: is the angular distance of a place north or south of the earth’s equator
- *Place Name*: is place’s description or name
- *Spare Time Order*: is a number that defines the order of spending spare time in different places

As described, each place includes coordinates and description, in order to measure distances among places. In addition, our *Travel Companion* when generate routes should know the places that are preferred for spending waste time. For instance, if we prefer wasting spare time at an airport rather than at a bus stop.
4.4.6 User’s Restrictions

For each user type we restrict traveler’s available transportation means. For instance, a disabled person may not be able to board on a bus, so if there is availability of another "more suitable" route we will recommend it. Figure 4.16 describes these restrictions.

Figure 4.15: Places Ontology

Figure 4.16: User Restrictions Ontology
4.5 REASONING CAPABILITIES

Common users could use any transportation means in order to navigate in an indoor place, however they do not need to use an airport car or a Wheelchair. Furthermore, Elderly can use an airport car, an escalator, elevator, or some stairs and Disabled an airport car, an elevator or a wheelchair for indoor navigation, however there restrictions are flexible and extensible. In case of another approach these dependencies could change, so our Travel Companion would detect these changes during the Travel Segment Generation and recommend different routes for the user. Such information described from our ontology and queries load such data depending the properties among them.

4.5 Reasoning Capabilities

ASP is especially well suited for problems that involve objects or states and relations between them. We developed the logic part of our approach using ASP language in order to guarantee that our method is strictly defined and our fundamental assumptions are preserved. There are well-known examples of symbolic computation whose implementation in other standard languages took tens of pages of indigestible code. When the same algorithms were implemented in ASP, the result was a crystal-clear program easily fitting on one page. Our system’s core consists of a Travel Segment Generator, an Alert Generator and a Personal Information Filter. The aim of these systems is to generate ordered actions and personalized information for the user, in order to provide the best Travel Plan.

Travel Segment Generator is able to create ordered atomic actions for each ticket. For instance, a trip that consists of two tickets, plane and bus, should recommend a different plan not only for each ticket, but also for each user category. Furthermore, we have defined constraint rules that actually limit user’s actions. Such restrictions could be in an airport, where our traveler can not reach his gate before crossing the luggage inspection area. Below is shown an example of such rules:

% Luggage Control is ahead Gate
:- goTo(luggageControl,S1), goTo(getToGate,S2), S1>S2.
Of course, some Travel Segments are optional, like Duty Free or Check In maybe optional if our traveler does not have any luggage to deliver.

*Alert Generator* is able to activate and send a notification to the traveler when necessary. For instance, if our traveler is approaching the luggage inspection area, whereas the next action is claiming his boarding pass from check in desk, our system is going to notify him about that.

When our traveler is located at a recreation area, like Duty Free, our system is able to make recommendation depending on his social media profile. For instance, if he likes "Starbucks" and he is located nearby, *Travel Companion* is going to recommend it for a coffee.

More examples of the way that such rules developed are described in detail in section 5.

### 4.6 Augmented Reality

In our approach, we consider that in the near future more and more virtual information are going to be embedded in our real world. The increased availability of smart phone and mobile gadgets has transformed the industry and will continue to enhance the ways in which travelers access information while traveling. Augmented reality has grown in popularity because of its enhanced mobile capabilities. A traveler would be able to see tremendous amount of information using his mobile phone and *Augmented Reality*.

Such information need to be filtered from Travel Companion in order to trigger user’s interest. Travelers prefer to see information that are more relevant to them, for our approach relevant to their social media profile. So, we can filter information about restaurants or cafes and make the best recommendation for our traveler. For example, if a user is located in an airport and does not know the suitable sign among "Check In", "All Gates" or "Luggage Control" scanning (image detection) these signs with smart-phone’s camera would be able to see embedded information that are relevant to user.

We propose using Vuforia for image detection. Vuforia is an Augmented Reality Software Development Kit for mobile devices that enables the creation of
Augmented Reality applications. It uses Computer Vision technology to recognize an image and then uses the data to compare the features in target image and the receiving frame from camera.
Chapter 5

Implementation

5.1 Description

*Travel Companion* is able to generate personalized Travel Plans for both indoor and outdoor navigation. Our system collects information using Google Maps API, Facebook API, information that are being collected using a smart phone’s camera and user’s personal information from Trip Ontology. Our Ontology was described in subsection 4.4, in order to provide the best route among the best places for our traveler.

Our approach aims to provide not only a path for reaching our traveler his destination, but also a Travel Plan among routes that will be pleasant. We used Augmented Reality in order not only to collect information, but also to enrich and filter such information from the environment, in order to be relevant for our traveler. We present below our approach for Travel Plan generation and describe the main features.

Figures 5.1, 5.2, 5.3, 5.4 and 5.5 shows the developed user interface. As shown in Figure 5.3, obligatory Travel Segments are bold and our user is able to see a deadline for each Travel Segment, the related place and Transportation Mean that is going to be used. Each Travel Segment is a button that allows user to view more details of his target, as shown in figure 5.4.
CHAPTER 5. IMPLEMENTATION

Figure 5.1: Login Page

Figure 5.2: List of Trips

Figure 5.3: Travel Plan
5.2 User Profile

As described in subsection 4.4.1, we are collecting information for a user, in order to provide the best plan. Furthermore, Facebook likes are being collected and all information will be converted to ASP predicates as shown below:

% User's Information
user(john).
hasAge(john, 69).
belongsToCategory(john, disabled).

% User's Luggage
carries(john, hand_luggage_john).
carries(john, heavy_luggage_john).

% User's Facebook Likes
likes(john, costa).
likes(john, starbucks).

As shown above, we define username, user’s category, our traveler’s luggage and his Facebook preferences.

General rules that define user’s category and condition are described below:

\[
\text{bagType(Bag, heavy) :- hasWeight(Bag, Weight), Weight>8.}
\]

\[
\text{bagType(Bag, heavy) :- containsPet(Bag).}
\]

\[
\text{bagType(Bag, hand) :- user(User), carries(User, Bag), not bagType(Bag, heavy).}
\]

\[
\text{belongsToCategory(User, elderly) :- hasAge(Num), user(User), Num>67.}
\]

5.3 Ticket Generator

Often, travelers only have some of the tickets necessary for them to reach their destination. For instance, if you want to travel from Athens to London you should hold your air ticket. However, when traveler reaches London, he is going to need a train ticket or a bus (any public transportation). So our system needs an effective way to provide directions for these tickets. To help the traveler in this task, our system used Google Maps API to load such routes.

Our approach exploits Google Maps API in order to load any possible public transportation (where available) for a specific route. A trip consists of one or more tickets, which are having a source and a destination place. Each place has its coordinates so we can easily calculate distances. If there are no Public Transportation from Google Maps API we recommend a taxi for this route. In case that we have to deal with elderly or disabled, we toggle options in order to have as fewer transfers
as possible, if time limits allow it. The property *Spare Time Order* classifies our preferred places for spare activities. For instance, if a user is located at a train station and consequently we need to direct him at an airport, we will choose a route that it will provide more free time in the second place than in a train station.

Briefly, there are some steps that need to be followed:

1. Load a trip’s tickets from Travel Companion’s knowledge base.

2. Parse each ticket and check the distance among the destinations of the current ticket and the source of the next one.

3. Load user’s property *Max Walking Distance* from our knowledge base.

4. If there is distance in step 2 greater than *Max Walking Distance*, then need to load a route for our user.

5. If our user is elderly or disabled the parameter "fewer transfers" is activated, a parameter which indicates that the calculated route should prefer a limited number of transfers.

6. Load from our RDF ontology-based data-set the order of public transportation, that indicates the order of use of a public transportation.

7. Load from our RDF ontology-based data-set the property of *Spare Time Order* that indicates places that we should prefer user spending more of his spare time.

8. Load from Google maps the first root that indicated the step 7, if there are more than once choose from step 5 ordering.

9. If the missing route that arises includes more than one tickets, store them back to RDF knowledge-base as "not available tickets".
The Java-like algorithm below describes the process of ticket generation:

```java
//Any Load method loads data from our Trip Ontology
//Returns a List with all necessary tickets
Algorithm GenerateTickets(Trip curTrip)
{
    //Information comes from Knowledge Base
    tickets = LoadAllTickets(curTrip);
    user = LoadTripOwner(curTrip);
    for(Ticket curTicket : tickets)
    {
        source = curTicket.GetDestinationPlace();
        destination = null;
        if(tickets.hasNext())
        {
            destination = tickets.Next().GetSourcePlace();
        }
        else
        {
            destination = curTrip.GetDestinationPlace();
        }
        // Check distance based on coordinates
        distance = CalculateDistanceAmong(source, destination);
        userDistance = user.LoadMaxWalkingDistance();

        if(distance > userDistance)
        {
            newTicket = GetFriendlyRouteFromGoogleMaps(source, destination);
            tickets.add(newTicket);
        }
    }
}
```
5.4 Travel Segment Generator

In order to provide the best Travel Plan for the user we have designed a mechanism, using ASP, in order to generate Travel Segments, described in subsection 4.4.4, in a meaningful sequence. Except for ordering these segments, it is important to define whether these segments are obligatory or not. For instance, leisure activities, like Duty Free, have been defined as optional.

We define obligatory Travel Segments for airports using rules as shown below:

\[
\text{shouldBeVisited}(U, \text{getToGate}, \text{yes}) :\text{- user}(U).
\]

So, any user should visit his gate, when he is going to have an air ticket.

Furthermore, some Travel Segments should be obligatory, when something occurs. For instance, if a user has delivered a Check in luggage, we should have generated a Travel Segment that directs him to the luggage claim area, after getting off the plane.

Such rules are described below:

\[
\text{shouldBeVisited}(U, \text{claimHeavyLuggage}, \text{yes}) :\text{- user}(U), \text{carries}(U, \text{Bag}), \text{bagType}(\text{Bag}, \text{heavy}).
\]

\[
\text{shouldBeVisited}(U, \text{claimHeavyLuggage}, \text{no}) :\text{- user}(U), \text{not shouldBeVisited}(U, \text{claimHeavyLuggage}, \text{yes}).
\]

Any user should visit luggage claim area if and only if has at least one luggage that belongs to him.

In addition, Travel Segments are obligatory when our user is an elderly or disabled. For example, such people may need help in an airport so calling someone from airport service is necessary.
Below is shown an example of such rules:

```prolog
% If user is an elderly, he should visit contactAirportService
shouldBeVisited(U, contactAirportService, yes) :-
    belongsToCategory(U, elderly).

% If user is not an elderly, he should not visit contactAirportService
shouldBeVisited(U, contactAirportService, no) :- user(U), not
    shouldBeVisited(U, contactAirportService, yes).
```

Any user should visit luggage claim area if and only if is an elderly. It is the same rule for disabled.

The rest Travel Segments are considered to be optional and described below:

```prolog
% Optional Travel Segments
shouldBeVisited(U, TrSg, maybe) :- travelSegmentType(TrSg), user(U), not
    shouldBeVisited(U, TrSg, yes), not shouldBeVisited(U, TrSg, no).
```

Also, getting the Travel Segments in partial order we use the rule below:

```prolog
% Max number of steps needed
steps(N) :- N = \#count{TrvSgm: travelSegmentType(TrvSgm), user(U),
    not shouldBeVisited(U, TrvSgm, no)}.
num(1..N) :- steps(N).

% Generate all possible combinations of steps
\{goTo(TrvSgm, Step):num(Step)\}1 :- travelSegmentType(TrvSgm), user(U),
    not shouldBeVisited(U, TrvSgm, no).
```

Using these rules, we first define the maximum number of steps that required and then all possible combinations of steps are generated.

Last but not least, these generated plans need to be ordered. For instance, we should guarantee that a user will not reach a Gate at an airport before visiting Check In.
5.4. TRAVEL SEGMENT GENERATOR

Constraints for air tickets are described below:

% Order Segments based on these limitations for airports

:- goTo(checkIn,S1), goTo(getToGate,S2), S1>S2.
:- goTo(contactAirportService,S1), goTo(checkIn,S2), S1>S2.
:- goTo(checkIn,S1), goTo(dropHeavyLuggage,S2), S1>S2.
:- goTo(checkIn,S1), goTo(luggageControl,S2), S1>S2.
:- goTo(dropHeavyLuggage,S1), goTo(luggageControl,S2), S1>S2.
:- goTo(luggageControl,S1), goTo(gotoDutyFree,S2), S1>S2.
:- goTo(luggageControl,S1), goTo(getToGate,S2), S1>S2.
:- goTo(gotoDutyFree,S1), goTo(getToGate,S2), S1>S2.
:- goTo(getToGate,S1), goTo(gotoSeat,S2), S1>S2.
:- goTo(gotoDutyFree,S1), goTo(gotoSeat,S2), S1>S2.
:- goTo(gotoSeat,S1), goTo(storeHandLuggage,S2), S1>S2.
:- goTo(gotoSeat,S1), goTo(aircraftExit,S2), S1>S2.
:- goTo(storeHandLuggage,S1), goTo(contactFlightAttendant,S2), S1>S2.
:- goTo(gotoDutyFree,S1), goTo(contactFlightAttendant,S2), S1>S2.
:- goTo(contactFlightAttendant,S1), goTo(claimHandLuggage,S2), S1>S2.
:- goTo(claimHandLuggage,S1), goTo(aircraftExit,S2), S1>S2.
:- goTo(aircraftExit,S1), goTo(claimHeavyLuggage,S2), S1>S2.
:- goTo(claimHeavyLuggage,S1), goTo(airportExit,S2), S1>S2.
:- goTo(airportExit,S1), goTo(airportExit,S2), S1>S2.
:- goTo(TrvSgm,S1), goTo(airportExit,S2), S1>S2.

These constraints specifies what Travel Segment is ahead of another.

We have defined plans for stations too (bus, train or subway tickets), like the constraints below:

% Order Segments based on these limitations for stations

:- goTo(gotoStores,S1), goTo(startPoint,S2), S1>S2.
:- goTo(kiosk,S1), goTo(gotoStores,S2), S1>S2.
:- goTo(kiosk,S1), goTo(startPoint,S2), S1>S2.
CHAPTER 5. IMPLEMENTATION

:- goTo(startPoint,S1), goTo(destinationPoint,S2), S1>S2.
:- goTo(TrvSgm,S1), goTo(destinationPoint,S2), S1>S2.
:- goTo(kiosk,S2), goTo(TrvSgm,S1), S1<S2.

For any Public Transportation we have defined four Travel Segments:

- **Go to Kiosk:** it is obligatory if the user does not hold a ticket of this route
- **Stores:** it is optional and refers to spare activities, like shopping
- **Start Line:** it is obligatory and refers to the place that traveler need to move in order to board on a Bus, Train, Subway
- **Destination Line:** it is obligatory and refers to the place that traveler need to get off

For Airports we have defined thirteen Travel Segments:

- **Check In:** it is obligatory for any user
- **Duty Free:** it is optional and refers to spare activities, like shopping
- **Claim Hand Luggage:** it is obligatory if the user carries a cabin luggage
- **Claim Heavy Luggage:** it is obligatory if the user carries a Check In luggage
- **Drop Hand Luggage:** it is obligatory if traveler carries a Hand Luggage
- **Drop Heavy Luggage:** it is obligatory if traveler carries a Check in Luggage
- **Contact Airport Service:** it is obligatory for elderly and disabled, in order to ask for help
- **Contact Flight Attendant:** it is obligatory for elderly and disabled, in order to ask for help
- **Get to Gate:** it is obligatory for any user
- **Airport Exit:** it is obligatory if the following ticket is not an air ticket that departs from the same airport
5.5 Deadline Generator

As we have already described in Section 4.4 we designed a Trip Ontology that describes in a more expressive way our form of data. In order to generate a deadline for a Travel Segment we need to take into account much information. Each user category may need different time duration in order to perform a task. For instance, an elderly may need a few minutes more or less in order to reach his Gate at an airport. Furthermore, carrying a hand luggage maybe it causes a delay. Some airlines recommend to passengers to be located at the airport at least three hours before boarding time.

When we generate Deadlines, we take into account many different restrictions and delays. As mentioned in subsection 4.4.3 we have proposed a property Time Before Departure, that defines the minutes that a user need to be located in the source place, before departing. In addition, in subsection 4.4.1 we had proposed the properties Luggage Extra Time and User Extra Time. They describe the time in minutes that need to be considered for each segment, when a user carries a luggage or/and belongs to a category that need more time to perform a Travel Segment. Finally each Travel Segment has a duration.

After generating all Travel Segments as described in section 5.4, each portion of Travel Segments belongs to a specific ticket which has Departure Date and Time and Arrival Date and Time. So, it means that Travel Segments that need to be performed before departure, need to have an earlier deadline than the rest.

The Travel Segment’s deadline is described below: First, we use the property Time Before Departure and we subtract this duration in minutes from departure.
date-time of out ticket. This is the date and time that we activate the first segment of a trip. Then for each segment we add to this date and time the Luggage Extra Time, if available, User Extra Time and finally, Has Duration which is different for each Travel Segment. For Travel Segments that need to be performed after departure we further add ticket’s duration.

Route begins...
Check In deadline = Time Before Departure + Check In Duration + Luggage Extra Time + User Extra Time

After Landing...
Luggage Claim Deadline = Arrival Time + Luggage Claim Duration + Luggage Extra Time + User Extra Time

5.6 Alerts

Alert is machine-to-person communication that is important or time sensitive. When a user receives an alert it is easy and quick to see, understand, and act in a particular situation. We generate it when a Travel Segment is about to expire and the traveler needs it as soon as possible.

Using Declarative Reasoning, via Answer Set Programming, we propose specific rules that activate an alert. For instance, if a traveler is located in an airport and looking for his Gate, he can use his smart-phone’s camera and detect signs around him. If he scans a relevant sign, like "Gates", Travel Companion is going to provide an alert for following or not this sign.

An example of such alerts is shown below:

% If sign matches with next pending Travel Segment
showToUser(U, comesCloserSign) :- user(U), goTo(Segment, MinOrd, pending),
    MinOrd = \#min{ Order : goTo(S, Order, pending) },
    caughtSign(U, Segment).

showToUser(U, walksAway) :- user(U), not showToUser(U, comesCloserSign),
5.7. SIGN DETECTION USING AUGMENTED REALITY

Consequently, we select the next Travel Segment and check if it is identified with our sign. If it is true then we activate an alert that informs user to follow it, else we activate an alert that informs him that this sign may not be relevant.

Another alert may be activated for free time activities, like Restaurants and Cafes. For instance, if a user is located at Starbucks, in Duty Free, and we match this information with Facebook’s likes, then we provide a recommendation for him to have a coffee.

Rules that activate personal preferences for our traveler is described below:

\[
\text{showToUser}(U, \text{preferenceAlert}) :- \text{likes}(U, \text{Preference}), \text{caughtPreference}(U, \text{Preference}), \text{user}(U).
\]

5.7 Sign Detection using Augmented Reality

There are travelers that do not know the local habits or the language of their destination country language. For instance, if a Greek visits Dubai, he may see Arabic signs at an airport, that is not able to translate. Furthermore, if our traveler is a disabled system should provide assistance avoid using stairs. For example, if there are two signs about "Check In" the first one is using escalator, whereas the second one using elevator. Travel Companion provides the best option for the user.

We previously presented in section 5.6 alerts that are shown to the user during a sign detection. Let us present now how do we detect these signs. Since our approach runs on a mobile device we used Vuforia for Image Detection.

Vuforia first detects "feature points" in a target image and then uses the data to compare the features in target image and the receiving frame from camera. If the feature points match, then detected image’s meta-data are returned to the user for possible alert detection (using rules like 5.6).
Chapter 6

Use Cases

In this section we propose some use cases in detail, in order to demonstrate our system’s innovations. First, we are going to describe the scenario and then some features of the system.

In order to add a new trip, our traveler should scan his available tickets and then set his source and destination place. After that Travel Companion is going to follow the process described in previous sections and generate a Travel Plan.

6.1 A teenager travels from Heraklion to Athens

**Scenario:** Maria is a seventeen years old young woman that lives in Heraklion. She is going to visit her uncle in Athens, but she has never been in an airport again. She is carrying just a Hand Luggage and plans to use Travel Companion in order to help her make it to Athens safely.

Maria starts scanning his available tickets, which are shown below ordered by time:

<table>
<thead>
<tr>
<th>No</th>
<th>Type</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bus Ticket</td>
<td>Bus Station</td>
<td>Heraklion International</td>
</tr>
<tr>
<td>2</td>
<td>Air Ticket</td>
<td>Heraklion International</td>
<td>Athens International</td>
</tr>
</tbody>
</table>

Table 6.1: Available Tickets of Athens’s trip

In order to set source and destination places for the trip (Astoria to Athens
International Airport), we use an additional ticket during scanning. We used it in order to check if our traveler is going to need more tickets than the available during her trip.

After scanning the tickets, using Vuforia, Travel Companion starts checking the trip for any missing tickets that need to be bought during the trip. Such tickets are going to be chosen very carefully depending on user’s category and profile loaded from our Trip Ontology. Consequently, our reasoner is able to generate the Travel Segments for each ticket. Finally, Java will set a deadline (time-stamp) for each Travel Segment combined with information from Trip Ontology and user’s profile.

As shown in figure 6.1 Maria has to use a bus and an airplane in order to get in Athens.

![Figure 6.1: Trip from Heraklion to Athens](image)

Now, Maria’s trip is about to begin. Travel Companion provides Travel Segments for bus ticket and air ticket as shown below:

- **Bus Ticket**
  1. *Stores*: Spend some spare time during waiting time for the bus
  2. *Start Point*: Place or Stop of departure
  3. *Destination Point*: Place that you need to get off the bus

- **Air Ticket**
  1. *Check In*: Place that need to claim your boarding pass
  2. *Luggage Inspection*: Place that police inspects your stuff, for dangerous products
  3. *Duty Free*: Place with shops in airports
6.2. AN ELDERLY TRAVELS FROM EDINBURGH TO MELBOURNE

4. *Get to Gate*: Place where the plane is located in order to board

5. *Take your Seat*: Find your seat number

6. *Aircraft Exit*: Get off the plane

7. *Airport Exit*: Get off the airport

Maria starts her trip with the first ticket, which is a city bus from bus station to Heraklion International airport. When she gets to the airport she checks Travel Companion and updates *Travel Segments* that has been successfully performed.

After that she approaches the airport on foot and realizes that she needs to find "Check In". Fortunately, Heraklion’s airport is a small airport so she finds it easily. Then, she decides to get to the gate in order to stay there until she boards on the plane, surfing the internet. However, Travel Companion instructs her to pass through the luggage inspection area, so she follows the instructions and then she gets to her gate, skipping optional Travel Segment "Duty Free". A few minutes later she gets on board and then takes off Heraklion. Finally, half an hour later she lands in Athens International Airport, where her uncle was waiting for her.

6.2 An elderly travels from Edinburgh to Melbourne

**Scenario:** John is a 70 years old elderly man that lives in Mansfield Traquair, in Edinburgh. Last year, his grandson relocated in Melbourne, near the Flagstaff Station, in order to study Computer Science. John decided to visit his grandson in Australia, but he is not familiar with traveling and has never been in Australia, so he feels a little bit insecure.

John believes that the application *Travel Companion* will help him to reach his destination, without having any trouble. His grandson, in order to help him, booked for him some tickets online, but John is going to need some more, because some tickets were not available online. Two days before departure John collects his stuff in a heavy check in luggage and in a small cabin luggage. After that he opens the Travel Companion and starts scanning his tickets.

John starts scanning his tickets, that are shown below ordered by time:
Chapter 6. Use Cases

<table>
<thead>
<tr>
<th>No</th>
<th>Type</th>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bus Ticket</td>
<td>Mansfield Traquair</td>
<td>Edinburgh Waverley</td>
</tr>
<tr>
<td>2</td>
<td>Train Ticket</td>
<td>Edinburgh Waverley</td>
<td>King’s Cross St. Pancras</td>
</tr>
<tr>
<td>3</td>
<td>Air Ticket</td>
<td>London Gatwick</td>
<td>Dubai International</td>
</tr>
<tr>
<td>4</td>
<td>Air Ticket</td>
<td>Dubai International</td>
<td>Melbourne International</td>
</tr>
</tbody>
</table>

Table 6.2: Available Tickets of Melbourne’s trip

These tickets include information described in our Trip Ontology, as mentioned in previous sections. John does not have any way to get from King’s Cross St. Pancras Station to London Gatwick and from Melbourne International Airport to Flagstaff Station. Travel Companion should find him the best routes in order to help John reach his destination. As shown in figure 6.2 John has to change many different Transportation Means in order to reach his destination.

Our system loads all the available Transportation Means for the missing routes using the proposed method and information from our Trip Ontology. After that a trip has been created including all tickets shown in figure 6.3. Some of them, have been loaded from Google Maps, whereas the rest are already owned by John.

It is time now for John to begin his trip to Australia. He opens the application and can scroll a list with all Travel Segments that he needs to perform.
6.2. AN ELDERLY TRAVELS FROM EDINBURGH TO MELBOURNE

It is shown below the Travel Segments for the bus ticket from Mansfield Traquair to Edinburgh Waverley and the train ticket from Edinburgh Waverley to King’s Cross st. Pancras:

- Bus Ticket

  1. Stores: Any stores near the Mansfield’s stop, during waiting time
  2. Start Point: The Mansfield’s stop, where the bus can be found
  3. Destination Point: The Edinburgh Waverley station

- Train Ticket

  1. Stores: Any stores near the Edinburgh Waverley’s stop, during waiting time
  2. Start Point: The Edinburgh Waverley’s platform, where the train can be found
  3. Destination Point: The King’s Cross st. Pancras station

Then the system generated two more tickets from King’s Cross st. Pancras to Victoria Station, using subway and from Victoria Station to Gatwick airport:

- Subway Ticket

  1. Ticket Kiosk: A place where a ticket to Victoria station should be bought
  2. Stores: Any stores near the King’s Cross st. Pancras’ station, during waiting time
  3. Start Point: The King’s Cross st. Pancras’ platform, where the subway can be found
  4. Destination Point: The Victoria’s station platform

- Train Ticket

  1. Ticket Kiosk: A place where a ticket to Gatwick station should be bought
2. **Stores**: Any stores near the Victoria’s station, during waiting time

3. **Start Point**: The Victoria’s station platform, where the train can be found

4. **Destination Point**: The Gatwick stop near the airport

After reaching Gatwick airport John has to perform the Travel Segments for two airport tickets that are shown below:

1. **Contact Airport Service**: Call for assistance in Gatwick

2. **Check In**: Place that need to claim your boarding pass

3. **Heavy Luggage Drop**: Leave your luggage to Check In desk

4. **Luggage Inspection**: Place that police inspects your stuff, for dangerous products

5. **Duty Free**: Place with shops in airports

6. **Get to Gate**: Place where the plane is located in order to board

7. **Take your Seat**: Find your seat number

8. **Contact Flight Attendant**: Call the flight attendant for assistance

9. **Aircraft Exit**: Get off the plane in Dubai

10. **Claim Heavy Luggage**: Claim your luggage from the specific area

11. **Contact Airport Service**: Call for assistance in Dubai

12. **Check In**: Place that need to claim your boarding pass

13. **Heavy Luggage Drop**: Leave your luggage to Check In desk

14. **Luggage Inspection**: Place that police inspects your stuff, for dangerous products

15. **Duty Free**: Place with shops in airports
6.2. AN ELDERLY TRAVELS FROM EDINBURGH TO MELBOURNE

16. Get to Gate: Place where the plane is located in order to board

17. Take your Seat: Find your seat number

18. Contact Flight Attendant: Call the flight attendant for assistance

19. Aircraft Exit: Get off the plane

20. Claim Heavy Luggage: Claim your luggage from the specific area

21. Airport Exit: Get out of Melbourne’s airport

Finally the system generates a bus ticket in order to get to Flagstaff station. Generated Travel Segments are shown below:

1. Ticket Kiosk: A place where a ticket to Flagstaff station should be bought

2. Stores: Any stores near the Melbourne’s station, during waiting time

3. Start Point: The bus stop, where the bus to Flagstaff can be found

4. Destination Point: The Flagstaff stop (destination)

Afterwards, John follows the Travel Companion instructions and gets the bus from Mansfield Traquair and the train from Edinburgh Waverley, whereby he reaches King’s Cross st. Pancras. Now, he needs to find a way to get to Gatwick airport, and Travel Companion recommends buying a subway ticket to Victoria station. As long as, we avoid walking long distances and using bus, it would be better for John to get to the airport through Victoria Station. He buys the ticket and finally reaches Victoria Station, where he needs to buy a train ticket to London Gatwick airport.

A few hours later he reaches Gatwick and does not know what to do. He checks the Travel Companion and he realizes that he needs to contact for airport service in order to request help. They provided him a wheelchair in order to avoid walking through the Gate. So he checks Travel Companion and realizes that "Check In" is the next place he needs to reach, he sees a sign that instructs him to turn left. Fortunately, that was easy, he found it and dropped his luggage there.
CHAPTER 6. USE CASES

Afterwards, he loves shopping and tries to get to Duty Free, but Travel Companion’s instructions seems to force his passing through the Luggage inspection area and then Duty Free. He spends walks through the stores and suddenly an alert is activated "You need to get to your gate for boarding". He leaves Duty Free and reaches his Gate where a flight attendant helps him to board on an Airbus A380 and takes off.

Now is time to relax until landing in Dubai International airport, where he has a self-transfer flight and need to be quick for his next flight. When he lands in Dubai he immediately gets to Luggage claim area following the passengers and he needs to find Check In of Emirates in order to deliver his luggage for the next flight. He is able to see some signs but they are written in Arabic, so he thinks the Travel Companion will assist him to follow the right sign.

John tries to use the Augmented Reality interface and scans the sign in figure 6.4 that directs user with the alert that says "This sign may not be relevant", and then the sign in figure 6.5 that instruct him to follow it.

![Figure 6.4: All Gates](image1.png)  ![Figure 6.5: Check In](image2.png)

Later, he reaches "Check In" leaves his luggage and looks for the gate in order to get as soon as possible. When he gets there he updates Travel Companion with "Get to Gate" Travel Segment completed. However, the system provides an alert to him that there are other obligatory Travel Segments before reaching Gate. So, he checks them and board on his last plane. He is taking a rest in the Airbus A380 because it is a long hour flight until his landing in Melbourne.

When he lands he successfully claims his luggage following the signs and check the Travel Companion that instructs him to buy a bus ticket that will take him direct to Flagstaff Station. Finally, he reaches his final destination after so many hours of trip, where he meets his grandson after a long time.

We should mention that in a real scenario some of these Travel Segments would be executed by default by airlines, airports, or stations. For instance, when John
performs "Check In" our system should be updated automatically, as he should have leave his luggage. Furthermore, when flight attendant scans the boarding pass in the Gate, "Get to Gate" should be completed automatically.
Chapter 7

Conclusions and Future Work

As the volume of travelers is continuously increasing, the need for managing and assisting them becomes all the more important. When passengers have a connecting flight in a foreign place they may get lost in large airports. Furthermore, they often need additional tickets when they travel abroad, except for flight tickets that have been booked beforehand. Our work addresses such problems, by proposing Travel Companion to assist them. Travel Companion is able to use information from our Trip Ontology and Facebook, in order to generate a personalized Travel Plan that assists travelers during their trips.

The next step for the Travel Companion will be to proceed to the Pilot Phase, in order to check the usefulness of our system. Additional development of the system is planned in order to provide further assistance in case of missing a bus, train or subway. We are going to re-plan the trip and provide alternative routes or even to provide the functionality of low cost routes. For instance, user could set a budget that is able to pay for a route and Travel Companion should find a solution for him.

Additional information can be found from other resources, like Google Places, or indoor maps that will assist further during indoor navigation in large airports or stations.

In the next steps, we are going to extend Travel Companion for more people categories. We will integrate a primary image-based detection system to assist
visually impaired people to independently travel and obtain the route information at a station or airport. Furthermore, we will extend our approach developing an interface that allows caretakers to track the position and receive alerts in case of anything fails, according to the Generated Plan.
Bibliography


Appendix A

Documentation

A.1 Environment

We used Java, Answer Set Programming, and Blazegraph for back-end development and Unity engine for front-end development. Below we express the reason we developed Travel Companion using these technologies:

- **Answer Set Programming**: offers an expressive way to solve logic problems.

- **Java**: is a highly portable language as it must be executed through a cross-platform compatible Java Virtual Machine (JVM). So during development we could focus on programming instead of worrying about the little details many consider both tedious and difficult about memory management etc.

- **Unity**: is easy to use, free and multi-platform, developing for one platform you can easily take your app to the rest. It is compatible with Vuforia, an augmented reality APK that we used in order to detect specific images.

We present below the versions that have been used for each technology:

<table>
<thead>
<tr>
<th>Technology</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clingo</td>
<td>4.5.4</td>
</tr>
<tr>
<td>Blazegraph</td>
<td>2.1.1</td>
</tr>
<tr>
<td>Unity</td>
<td>5.3</td>
</tr>
<tr>
<td>Java</td>
<td>1.8</td>
</tr>
<tr>
<td>Vuforia</td>
<td>5.0.10</td>
</tr>
<tr>
<td>Tomcat</td>
<td>8</td>
</tr>
</tbody>
</table>

Table A.1: Development Versions
A.2 Installation

Travel Companion has been tested using Android tablet and back-end development using Mac OS, but it works perfectly to the rest of the operating systems. After installing and running successfully Server, you can run your Client.

A.2.1 Server

First, use a pc for downloading Netbeans, Clingo, Blazegraph, Java, Tomcat. Install Java, Netbeans and Tomcat using their instructions.

Using Mac OS, Linux or Windows pc that supports anything in A.1, follows the steps below:

1. Run Blazegraph
2. Copy the provided service URL of Blazegraph
3. Import server’s project zip in Netbeans (TravelCompanionIslForth)
4. Open /WEB-INF/props.properties file and set the Blazegraph URL from step 2. Be careful, url should not end with a slash ("/").
5. Find the project "TravelCompanionIslForth" in Netbeans folder, you should have in the same directory the "TravelCompanionIslForthFileSystem" in order to be visible to Java
6. Place the suitable executable file of Clingo to /TravelCompanionIslForthFileSystem/clingo/ directory depending your Operating System
7. Right click to project and press "Clean and Build" and wait a few seconds
8. Right click to project and press "Run" and choose your favourite Server (I used both Glassfish 4.1.1 and Tomcat 8)

A.2.2 Client

For further development install the suitable version of Unity described in A.1 and import Client project folder.

In order to run it use an Android tablet or smartphone and follow the steps below:

1. Install the TravelCompanion.apk
2. Open the application and from the upper right side click on settings and set server’s IP address
3. Save and go back for log in (john, nick and prodromos are available user ids)
A.3 Instructions

User can add a new trip and the system provides him a list of Travel Segments. Afterwards user can update the state of each Travel Segment by ticking the check-box or undo his choice. Furthermore by ticking the camera button can open an Augmented Reality interface and scan signs around him. When user scans signs he is able to see notifications about these signs that need to follow.

A.4 Code Structure

There are some source packages under TravelCompanionIslForth project:

- Utilities: Anything relates to SPARQL (Queries and Updates), Google Maps, Facebook
- Clingo Manager: Anything relates to Clingo execution
- Web Sockets: The way to send messages to Client
- Blazegraph: Any utility about Blazegraph (Load, Update, Delete and Save triples)
- Parser: In order to parse results from Clingo, Facebook, Google
- Controller: Integration of any proposed technology
- Services: Requests the received from Client are triggered here
- The rest are used to convert Java objects to Json objects