OPTORER: A Dynamic Routing and Touring Service for Indoors and Outdoor Tours

Constantinos Vassilakis , Maria Polychronaki , Dimosthenis Margaritis , Dimitrios G. Kogias , ELENI LELIGKOU *

Posted Date: 7 February 2024

doi: 10.20944/preprints202402.0404.v1

Keywords: Routing service; UX optimization; Thematic tours marketplace; Content Creation; Public Safety and Wellbeing; Dynamic Routing; Smartphones; Wearables

Article

OPTORER: A Dynamic Routing and Touring Service for Indoors and Outdoor Tours

Constantinos Vassilakis [1], Maria Polychronaki [1], Dimosthenis Margaritis [2], Dimitrios G. Kogias [1,*] and Eleni-Aikaterini Leligkou [2]

[1] Dept. of Electrical and Electronics Engineering, University of West Attica; { convassilakis, m.polychronaki, dimikog}@uniwa.gr
[2] Dept. of Industrial Design and Production Engineering, University of West Attica; {auto45184, e.leligkou}@uniwa.gr
* Correspondence: dimikog@uniwa.gr

Abstract: This paper introduces a new routing and touring service both for outdoor and indoor places of touristic and cultural interest designed to be used in the wider area of Attica, Greece. This service is the result of the work done in OPTORER project and it aspires to offer a range of innovative and thematic routes to several specified points of interest in the selected area of Attica, encouraging the combination of indoors and outdoors routes in a single tour. The aim is to optimize the user experience while promoting specific, user-centric features with safety and social welfare being a priority for every designed tour resulting in enhancing the touristic experience in the area. Using a common smartphone device, as well as common wearable devices (i.e., smartwatches), the OPTORER service will provide an end-to-end solution by developing the algorithms and end-user applications, together with an orchestration platform responsible for managing, operating and executing the service that produces and presents to the end user results derived from solving dynamically complex optimization problems.

Keywords: routing service; UX optimization; thematic tours marketplace; content creation; public safety and wellbeing; dynamic routing; smartphones; wearables

1. Introduction

Tourism is one of the main pillars of the global economy. At the same time, social distancing has recently become one of the most pressing social problems due to the COVID-19 pandemic, since it introduced periods of mandatory confinement which resulted in a social chain reaction in terms of travel, whether for entertainment or other reasons. And while tourism has been the most stressed sector affected significantly by the pandemic, it has managed to recover and see exponential growth in the past 1-2 years.

This growth has been supported by technological advancements, as more and more attention is given to applications and systems which aim to bring the tourism sector (e.g., hotels, tours, museums) closer to the digital era, offering an enhanced experience to citizens, but also advantages to professionals. More specifically, in [1] the authors are focused on *'Smart Tourism'*, carrying out a thorough investigation of the technological developments in the field in the years 2013 - 2019. By examining more than 50 different bibliographic sources, they explain that during this time, specific technical terms have emerged, such as 'Smart Tourism' and 'Smart Tourism Destination', which express the citizens' need for technological support, elevating their tourism experience to another level, at each destination. Analyzing their findings in 12 technological pillars (e.g., Social Networks, Internet of Things and User Experience), they also list proposed Smart Tourism systems, where among them, a large percentage are navigation and destination recommendation applications (i.e., restaurants, museums, accommodation, etc.). An important role is played by the fact that these

 doi:10.20944/preprints202402.0404.v1

applications can be accessed by tourists through their mobile device, adding further immediacy and personalization to their experiences.

Additional findings that prove the above are reported in [2], where the authors in their attempt to classify the sectors in Smart Tourism research, they refer to numerous reports of Smart Tourism Technologies (STTs). These focus on optimized tourism experience, consumer satisfaction and behavioral mobilization. In fact, among the systems preferred by tourists, they cite as typical examples of Smart Tourism adoption, experience organization applications (e.g., determining routes, planning points of interest), but also handling procedures (e.g., completing payments, booking seats and tickets).

Finally, in [3], the authors are conducting research on the adoption of mobile technologies and applications in Smart Tourism, focused on the demand for Smart Tourism applications from the consumer side. Their results indicate that there is increased satisfaction as well as psychological motivation for tourists when they are given the opportunity to act through their mobile device and have as many services as possible available at their 'fingertips'. Especially attractive for tourists are applications related to accommodation and catering, applications with travel services (tickets, securing seats) and last but not least, applications for browsing and providing information about destinations.

OPTORER is a research project [4] that aims to create a platform to support and enhance the touristic experience initially in the region of Attica, Greece. To achieve this, OPTORER introduces *a novel routing and exploration service in outdoor and indoor areas* of touristic and cultural interest in the broad area of Attica. OPTORER's service falls under the category of Smart Tourism Technologies and Applications, covering a wide list of offered functions that are considered attractive to a tourist, while introducing several new, innovative features for user satisfaction and psychological motivation.

To achieve this, there are three main technical challenges that the OPTORER service manages to address and provide efficient solutions. These challenges are:

- *Indoor location and positioning*: GPS technology has made outdoor positioning very successful and widely supported. However, due to signal attenuation caused by building materials, indoor positioning systems cannot rely on this technology for efficient measurements. At the same time, many indoor positioning systems have been developed based on a wide range of technologies, including WLAN, infrared, ultrasonic, and Bluetooth [5]. Indoor location and positioning is made possible through the use of techniques such as trilateration, triangulation and fingerprinting [6], that calculate distance using the strength level of a signal received by a user device. The signal is generated by multiple radio beacons placed indoors, while relative position is decided based on the accumulated distance from the beacons. Bluetooth radio beacons covering indoor spaces can provide a low-cost, low-power solution, while a smartphone can act as a receiver. The accuracy achieved in positioning and tracking is critical for OPTORER service, [7].
- *Physical user state assessment:* The physical effort exerted, or the physical state of a user at a particular moment, can be assessed using algorithms that have as input heart rate monitoring data and/or cadence/pace measurement, obtained from sensors embedded in smartphones and wearable devices. However, accuracy in estimation is really a challenge. In addition, estimating user experience from this data is a step forward in this type of evaluation as it attempts to capture real-time user sentiment regarding user satisfaction, [8].
- *Dynamic Multi-Criteria Routing Optimization*: The exact solution (i.e., finding a globally optimal solution) of the routing optimization problem will make it impossible to provide near-real-time results, as the routing problem belongs to the class of optimization problems known as NP-Complete, which means that it is not possible to quickly find an optimal solution, as the complexity increases significantly as the destinations involved in routing increase, [9]. The need to develop efficient meta-heuristic algorithms to provide a near-optimal solution to the multi-criteria optimization problem, in near-real time, is a challenge that has been addressed in OPTORER application. Furthermore, due to the dynamic nature of a tour, the service is required

 doi:10.20944/preprints202402.0404.v1

to be able to partially solve the optimization problem when some of the variables change in order to provide updated results immediately, [9].

In general, another additional challenge is the study of how to scale the service to a large number of users, especially considering that complex algorithms that should be executed in real-time per user for a tour. This includes not only the demand for high-performance algorithms but, also, for highly efficient management of infrastructure resources.

For the rest of this paper, the structure of the work is the following: in **Section 2** details about the Design of the System are provided with emphasis on the main sub-systems that are needed, followed in **Section 3** with more details about the System Architecture and the solutions to the technical challenges discussed above. In **Section 4,** the results from the piloting events of the project, related to the system and measured module performance, are presented while in **Section 5** the conclusions and possible extensions of the work are discussed.

2. System Design

Solutions to the technical challenges presented above are key in order for OPTORER to be able to meet its main objectives. These objectives are:

- ✓ To offer a routing and tour planner service in outdoor/indoor places of touristic and cultural interest able to adjust to achieve specific purposes (personalized criteria and/or promoted purposes).
- ✓ To expand the number of tours indoors requiring only a low-cost initial investment from the operators of places of interest using BLE beacons.
- ✓ To assess the user experience and drive the dynamic adjustment of routing decisions along the tour.
- ✓ To provide the current state with the ability to ensure in real time the safety and well-being of citizens by communicating notifications or alerts to be taken into account in the dynamic routing decision process.

Taking under consideration the possible technical solutions that could be used in order to meet OPTORER's objectives, the system is designed to include four main subsystems:

1. User Applications subsystem
2. Development subsystem
3. Controller subsystem
4. Infrastructure subsystem

Each of these subsystems plays a crucial role in the performance of the overall OPTORER system. In Section 3, details about the interconnection of the subsystems is provided to describe the overall system architecture. Here, a description of the functionality that each one brings to the system is given.

2.1. Description of OPTORER's sub-systems

The "*User Applications*" subsystem consists of 2 applications: a *mobile application* that enables the end user to navigate and customize an offered thematic tour using their smartphone or setup a new tour by selecting favorite points of interest through the app. The guidance of the user will be continuous no matter if it is in an external or an internal space, as long as the latter supports the OPTORER service with the required infrastructure (as explained later). The mobile application also offers the *user experience assessment* functionality which will run continuously during the tour as long as the user wears a compatible wearable device (i.e., a smartwatch). This functionality collects data from the mobile phone sensors and the smartwatch in order to assess the physical state of the user. The heart rate information accurately provided by wearable devices is used while additionally the estimation of steps, which can be extracted both from the wearable device and from the mobile phone, is also monitored. The calculations are performed on the smartphone delivering fast estimation and notification delivery and the data are not stored, in respect of the GDPR rules in Europe. Any provided feedback / assessment from the user (e.g., the need to rest) will be delivered to the Controller subsystem for real-time, dynamic changes to the provided route with the aim of optimizing it. There

 doi:10.20944/preprints202402.0404.v1

is also a *web application* that is used to manage the users of the service but, also, to add and manage content that is offered. For example, for an indoors tour, the user can access the web application to draw the map of the interior place and, also to add important points of interest inside it. To allow such an action, the user should have a specific role. The role is assigned and monitored, also, by the web application. More details about the roles are provided in Sub-Section 2.2.

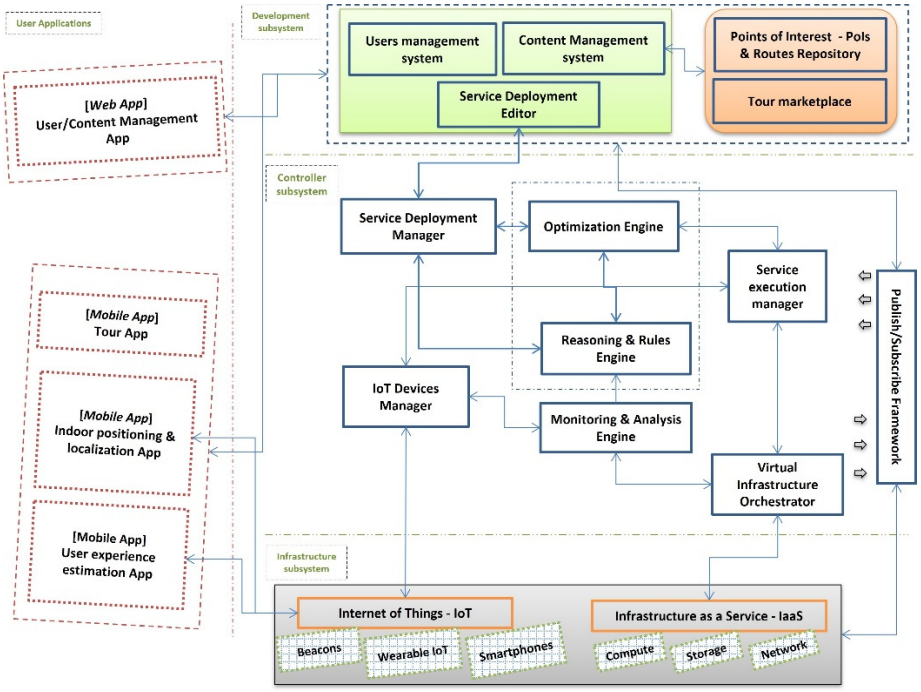

Figure 1. OPTORER Sub-Systems and their components**.**

The *"Development Subsystem"* implements *the management of users and content* of the service while, per user and selected tour, it configures the initial conditions and constraints of the routing problem. The solution to the routing problem is performed dynamically and according to the user's choices, the real-life reported changes on the conditions of the tour and the monitored assessed experience of the user. The routing problem is continuously resolved by the Controller Subsystem and the results are fueled to the mobile application to guide the user on his tour to and navigate him/her through a selected number of Points of Interest (PoIs). In general, the Development subsystem includes five components: the "*User Management System*" component that implements the functions offered through the Web application involving the management of users and the management of the subsystems. These functions are allowed for the user with the role of *Administrator*. Additionally, there is the "*Content Management System*" component that implements these functions that are offered through the Web application concerning the user role-based management of the content. These functions are available for the user with the role of *Navigator* or *Cartographer* or *Service Partner* or *Feedback Provider*. In the *"Repository of points of interest and routes*" component the processed content concerning Areas of Interest (AoIs) or Points of Interest and routes is made available at different levels of detail, as required per case (e.g., a full map of the AoI if it concerns an equipped interior with a series of radio beacons). The Content Management System and the Repository of PoIs are the components used to create and configure a structured tour that is made available in the *Tours Marketplace* component. Finally, the "*Service Deployment Editor*" component is responsible for initializing the routing problem with those data and constraints, as configured by the selected tour and/or the user's choices and preferences submitted by the User Application. It is the point of contact with the Mobile Application, both for guiding the user through a tour as a product of solving the routing problem in the Controller Subsystem, and for dynamically configuring the data and constraints governing the tour as a result of continuously evaluating user experience from the User

 doi:10.20944/preprints202402.0404.v1

Experience Assessment functionality, but also from changes in the surrounding conditions because of unforeseen events while touring.

The "*Controller Subsystem*" is at the heart of the OPTORER System. It manages and monitors all the devices and resources of the service, as well as the changing conditions that dynamically shape the routing in a tour, altering the parameters in the multifactor optimization problem that is being solved in this subsystem for each user and each tour. To deliver this performance, the Controller subsystem consists of several components. Starting with the "*Service Deployment Manager*" which is responsible to receive and forward the conditions, regarding each user's tour, of the route problem to the *"Reasoning & Rules Engine"* component. The "*Optimization Engine"* is the component that runs the code of the applied model for the optimization of the tour problem which uses selected meta-heuristic algorithms. The problem gets dynamically updated and enriched by the rules and constraints retrieved from the *Reasoning & Rules Engine* component. Additionally, the Reasoning & Rules Engine forms a collection of rules and restrictions which are stored in a structured way. Here, there is also the "*Internet of Things (IoT) Devices Manager"* which is responsible for constantly collecting and providing information related to and retrieved from the available IoT devices (e.g., radio beacons, wearable devices, smartphones) in the system. It is also responsible for their management. The data that are collected include information about the location of the user, by relating the geographic location of a mobile device (outdoors) or its location relative to a radio beacon in the case of an indoor location, as well as other information such as the user's current assessment of experience or, more fundamentally, its network connectivity. At the same time, there is the "*Virtual Infrastructure Orchestrator"* who is responsible for continuously monitoring of the System's needs for resources and adapts the offered resources depending on the demand. The last component is a publish – subscribe broker mechanism that exists to deliver notifications which are received from the IoT devices (e.g., alerts regarding the physical condition of the user that should trigger a dynamic recalculation of the touring route).

Finally, there is the "*Infrastructure Subsystem"* that includes all the actual IoT devices (e.g., Beacons, Smartphones and Smartwatches) and the Cloud Resources that belong to the Infrastructure as a Service (IaaS) solution.

2.2. Types of Users and Allowed actions

To complete the design of the OPTORER system, the different user roles and the actions each one of them is allowed to perform are presented in this sub-section.

The basic user roles are:

- *End User:* The end user of the service uses the Mobile Application to follow a selected tour (pre-planned or custom). Users of the Mobile Application are able to search among pre-planned tours filtered by the category of interest, the cost, or the available time. Additionally, the end users can create their own tour by setting the following details: the starting point, specific order for visiting the already registered PoIs, the endpoint and the preferred means of transportation. In this case, PoIs can be selected using a filter based on the category of interest. Also, the end users can use the mobile application to report an event at the point where they are, follow it by a short description of the event.
- *Administrator:* The Administrator can define technical parameters of the system, gain access to data recorded for security reasons (e.g., unsuccessful login attempts) and manage users (i.e., in terms of access role and status). There are also certain administrative functionalities that have specific roles assigned to them (i.e., administrator sub-roles). These roles are:
    - *Partner's Content Management*: Core Functionality of this role is to review the content/services that partners intend to include in the system. Examination of the content offered (e.g., texts and photos) is essential in online platforms to avoid malicious content and also to avoid listing mistakes by partners.
    - *Event Management*: Core Functionality of this role is to review data related to events reported by end users or other sources (communication streams). Reviewing events before

 doi:10.20944/preprints202402.0404.v1

communicating them to end users is essential to avoid false alarms, which can cause unnecessary panic.

- *Tour Creator:* The Tour Creator creates and configures a tour, defining the sequence of visits to Areas of Interest as well as the sequence of visits to Points of Interest within the selected Areas. The tour can combine both indoor and outdoor locations. Additionally, the tour creator enters content related to the Areas and Points of Interest while setting tour scheduling and possible stops/visits to points related to external Partners Provided Services. This way, the Tour Creator can enhance the end user's experience during the tour or can cover possible needs along the way.
- *Cartographer*: The Cartographer creates maps of indoor spaces and defines the location of indoor PoIs. Also, the Cartographer utilizes the required radio beacon infrastructure as well as the topographical mapping of the area to define a tour inside the indoors area. Additionally, Cartographer can add more detailed descriptions to outdoors PoI that can be included in tours combining the added indoor location.
- *External Service Provider*: The role of the External Service Provider can enter offering services, as well as adding points where these services can be found. These services can be included by each Tour Creator in any predesigned or for a custom tour. Given that the platform supports Business-to-Business (B2B) capabilities, a tour is enriched with offered services intended to be provided by third-party companies or freelancers. For example, at a museum a point of interest can be added pointing to a tour guide that offers guided tours. Accordingly, a travel agency could list its own recommended tour to be delivered through the OPTORER platform at a certain cost. A user can be added to this role by applying for registration by filling out a relative form.
- *Provider of feedback*: The Provider of feedback role has the ability to provide feedback in relation to the service, the tours as a whole or in relation to individual PoIs or professionals/services involved in the tour. The feedback can take the form of an evaluation but, also, about providing relevant content. Every user of the service who participated in an offered tour will be able to provide feedback.

Finally, a user may have more than one role after being assigned the corresponding access rights**.**

3. System Architecture

In the previous section, a presentation of the four main subsystems that were designed to form the OPTORER System took place. Here, a discussion about the resulted architecture of the OPTORER System will be presented, followed by a presentation of the solutions to the main (research) challenges that were selected and identification of the system component that performs the designed actions.

Figure 2 depicts the final design of the OPTORER System Architecture showing the components of each sub-system and the basic communications between them. Moving away from the design in Figure 1, the initially described sub-systems have received some minors adjustments and are highlighted by certain important tools, as seen in Figure 2. These tools and their interconnection with the rest of the OPTORER System are described here in detail for every subsystem:

- the "User Applications" subsystem consists of all the applications that a user of the system can access. There are three types of applications in general in OPTORER:
    - a web application which provides the authorized users (i.e., those with the role of Administrator, Tour Creator, Cartographer and External Service Provider) access to the Development subsystem. This access is provided through two different tools, the *OPTORER Admin Web Portal* tool and the *Cartographer* tool. The former allows for administration actions to take place along any content management actions (e.g., adding, removing or editing information for PoIs or AoIs) while the latter is the tool by which content managers can add the map of an indoor tour. This is done using the graphical environment available from the Cartographer tool in order to draw the floor plan of the interior space, as well as to register the points where the BLE Beacons have been placed in the respective map. The last step of the process includes setting up of Internal PoIs.

 doi:10.20944/preprints202402.0404.v1

- a mobile application, that is the main way in which the end user interacts with the OPTORER system and how (s)he can access and enjoy the functionalities and services offered by it. The mobile application has direct connectivity with all the other main tools in the other sub-systems (seen in Figure 2) including:
  a. Connection to the wearable device application based on the Bluetooth protocol and using the native interfaces offered by its operating system.
  b. Connection to the tools in the Development subsystem using the HTTP protocol and the available REST APIs, in order to obtain and configure the application relevant information for the user.
  c. Connection to the Routing Engine tool inside the Controller subsystem using the HTTP RESTful interfaces offered by the corresponding GraphHopper tool, [11].
  d. Connection to the EMQX MQTT Broker tool in the Infrastructure subsystem, a real-time communication infrastructure using an MQTT client and subscribing to the channels that offer information related to emergency events that can affect the user experience. Here there is a change in the initial design (Figure 1), as the MQTT (pub-sub) broker is moved to the Infrastructure subsystem instead of the Controller subsystem since the emphasis is given to the communication of the information, rather than the control and processing of it.
  e. Connection to the Indoor Location Infrastructure tool using the Bluetooth protocol for continuous scanning of the available BLE Beacons and real-time positioning of the user on the indoor map.
  f. Integration of Indoor Routing Engine functionality, using Dart language for smooth operation of the application. This tool is used to locate and position the user when (s)he moves in an indoor room. Also, the tool offers navigation using a pre-installed indoor map (by using the Cartographer tool).
- A smartwatch application (OPTORER Wearable Companion Application) that provides information to the mobile application related to the user's physical state. This is done locally by pairing the user's watch with their mobile device using the Bluetooth protocol. This information contributes to the user experience exclusively and only in real time during a visit, while at any time, the user can through the mobile application delete these measurements from the memory of the application.

- The "Development subsystem", or in other words the *Information Backend*, consists of two main tools (i.e., Vertoyo VDP and Vertoyo Database) that incorporate and deliver the actions and operations described in Section 2 for this subsystem. In more details, Vertoyo Digital Platform (or Vertoyo VDP) is a platform based on the Java programming language and can be used to support complex web portals and mobile applications through a variety of ready-to-use features such as managing user roles/privileges, entity management/registration. It comes with advanced support functions and management of business workflows (Business Process Management - BPM), task scheduling and can be easily interfaced with Push/Email notification services, as well as with third-party systems, exposing or consuming programming interfaces such as REST or SOAP APIs. VDP communicates directly with the Vertoyo Database tool, a PostgreSQL DB that implements a custom SQL schema to store data for: a) User Management - encrypted data for sensitive fields, b) Content Management - static content of the Web application (e.g., PoI or AoIs), c) Custom Entities Management – supporting data encryption. It is used to store the alerts coming from the MQTT pub-sub broker in OPTORER.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 February 2024 doi:10.20944/preprints202402.0404.v1

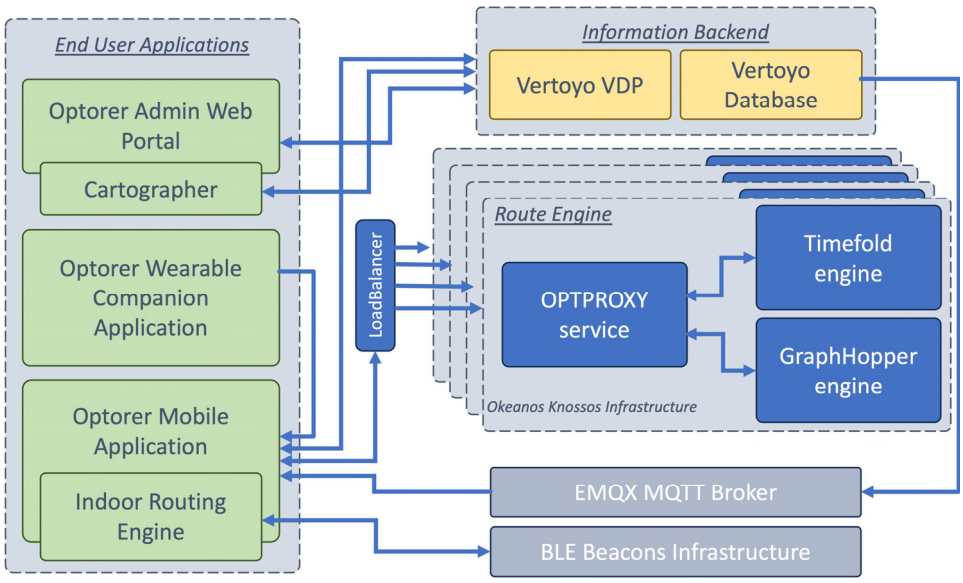

Figure 2. OPTORER final System Architecture.

The two tools inside the Development subsystem communicate with the web and mobile applications in the User Applications subsystem to give access to the back end to the certified users, and with the MQTT broker in the Infrastructure subsystem to receive updates that need to be stored in the system.

- The "Controller subsystem", which is now called the *Route Engine* component, is a complex routing module that consists of three tools: the *OPTPROXY Proxy Application*, the *Graphhopper Routing Engine* and the *Timefold Optimization Engine*. With the service implemented by the OPTPROXY proxy application, it is possible to communicate with the mobile application to submit the routing request and send a response to the user. Internally, the OPTROXY application takes care of communicating with the other two engine tools to return the optimal visit order if desired, as well as the optimal routing between them on the road network based on the user profile, the desired characteristics submitted by the user, and possible external constraints due to unforeseen events. In the final version, load scaling is achieved by installing the "Route Engine" as a set consisting of the proxy application, the Graphhopper routing engine and the Timefold optimization engine on multiple virtual machines (Virtual machines) in a virtual server infrastructure and sharing the load on them (round robin).
- The "Infrastructure subsystem" consists of two tools: the EMQX MQTT Broker and the BLE Beacons Infrastructure. The former is a real-time information communication tool, implemented by an EMQX Broker. Through this, all the necessary information of any emergency events confirmed by the admin portal are transmitted to the end user application on the mobile phone. In real time, this information is filtered based on the importance of the event, the current location and the status of the user. For smooth and optimal operation of the platform, the real-time communication infrastructure is connected to two of the other tools of the system:
  - The mobile application for end-users, for the dynamic and customized routing of the user, based on emergency events.
  - The Information backend to send real-time and continuously update the VDP database and the mobile application about the status of emergency events, during a browsing.

There is also the *BLE Beacons Infrastructure*, which is in fact independent of the rest of the system, while at the same time its communication is passive and is carried out exclusively by the end user's Mobile Application. The infrastructure consists entirely of the **BLE radio beacons**, which are installed indoors by the respective building managers. At the same time, with the help of the Cartographer tool during installation and in combination with the tools inside the Development subsystem, the user's mobile phone can carry out all the functionalities related to the real-time positioning and navigation of the user in the corresponding interiors. It should be noted that apart from a difference

 doi:10.20944/preprints202402.0404.v1

in the visualization of the room, the experience is the same both for indoors and outdoors tours, meaning that the selection of points and the routing offer the same experience to the end user.

3.1. Technical Challenges

Having discussed the OPTORER System Design and (final) Architecture, it is time to delve deeper into the research and implememtation decisions that were made to answer the technical challenges in OPTORER (see Section 1). The implementation of these solutions made it possible to meet the objectives of the project efficiently.

In this sub-section, the solutions to the three challenges that OPTORER faced are presented in detail.

3.1.1. Indoor Location and Positioning

OPTORER delivers a novel routing and exploration service that combines indoor and outdoor exploration in a single combined tour. While for the outdoors routing, the use of GPS has been recognized as the go-to solution, things are not so straightforward regarding indoors technology, where the solutions are under research and receive continuous updates.

Some of the more commonly used techniques for indoors location include:

- Wi-Fi fingerprinting
- Inertial Navigation
- Bluetooth Low Energy (BLE) beacons
- Visual-based Navigation
- Magnetic Field-based Navigation

For the most part, the above methods are quite different from each other, as they use completely different equipment as well as a wide class of algorithms to place an object/user in an interior space.

In [12], the authors examine the use of metrics such as Wi-Fi Received Signal Strength Indication, Wi-Fi Round Trip Time and visible signs, which are processed by trilateration, Fingerprinting with Neural Networks and Fingerprinting with Nearest Neighbor algorithm. Of the algorithms, the neural networks and Nearest Neighbor methods yielded maximum accuracy, with a small difference between them. Trilateration yielded an order of magnitude worse accuracy. A more detailed study on the trilateration method was produced in [13]. In this study, the authors describe the method of trilateration in detail. Next, they describe an experimental setup with three BLE transmitters in a room 9.0m long and 10.2m wide. Using a mobile device, they measured the error of the method at 163 points in the room. Their measurements were made with sample sizes of 1, 5, and 10 replicates and produced errors with means of 39.06m, 8.96m, and 7.97m and medians of 8.04, 5.54m, and 4.23m, respectively.

On the other hand, Bluetooth Low Energy (BLE) is a communication protocol for Wireless Personal Area Network technology, developed by the Bluetooth Special Interest Group (SIG). BLE, known as Bluetooth Smart, is part of the Bluetooth 4.0 standard, however, it has several additional functions compared to classic Bluetooth.

Indoor navigation using BLE radio beacons is an active area of research and development and is constantly evolving. BLE beacons are small battery-powered devices that transmit signals that can be received by mobile devices and have been used in a variety of applications, including indoor navigation.

Some of the current approaches to indoor navigation using BLE beacons include:

- *Fingerprinting [14]:* This method involves creating a database of signal fingerprints at known locations within a building and by using these fingerprints to estimate a user's location based on signals received from nearby radio beacons. This approach has proven efficient in many indoor environments, but can be computationally demanding, requiring frequent updates as the environment changes.
- *Trilateration* [15], [16]: This method involves using the signals from at least three radio beacons to triangulate a user's location. Free Space Path Loss (FSPL) formula is a popular method to be

 doi:10.20944/preprints202402.0404.v1

combined with trilateration since it converts the Received Signal Strength Indices (RSSI) to beaccon distance and then use this as input to the trilateration process to convdert distances from known beacon location to coordinates. This approach is less computationally demanding than signal fingerprinting, but may be less accurate in environments with obstacles or other sources of interference.

- *Machine learning* [17], [18]: This approach involves training machine learning algorithms to recognize patterns in signal data and use these patterns to estimate a user's location. This approach has the potential to be highly accurate and adaptable to changing environments, but requires large amounts of training data and can be sensitive to changes in the signal environment.

In summary, any approach to the indoor location problem must balance between accuracy and ease of implementation. At one end, there is the conventional solution of trilateration, which for two dimensions needs three transmitters and their coordinates. At the other extreme is the use of Fingerprinting, which requires a significant time investment to produce the measurements. In the case of data processing using neural networks, significant computing resources are also needed for their training.

A middle ground is demanded for OPTORER, as some solutions do not provide enough accuracy to be useful, while others are extremely difficult to implement in the real world, as the required measurements are extremely time-consuming and any change in the layout of the space requires repeating them from of zero.

The solution in OPTORER: In order to identify the optimal technique for OPTORER, a solution thath balances between accuracy and feasbility is followed, **the One Dimensional Fingerprinting (1FDP)**. Measurements that were conducted at the University of West Attica's Ancient Olive Grove Campus, Hall ZA216, a rectangular room measuring 8.1 over 8.02m with 3 beacons located in well-known positions have supported the efficient performance of the approach.

Details about the methodology of the proposed solution and evaluation results for the performance of 1FDP can be seen in [19].

Finally, it should be noted that for this service to be available to the end-user, a set of actions from the owner of the indoor space should take place in advance. First, the mapping of the indoor space should be designed using the Cartographer tool. There, the initial decisions to be made are the marking of spaces and the identification/assignment of the type of each space. The floor plan of each space is recreated through the repeating process of spaces marking by defining their shape, and placing inside the space of the corresponding emitters. After this step, the connections between the rooms are defined (e.g., mark any corridor) and the points of interest are also defined. The file then is uploaded to the Information Backend (see Figure 2) and is ready to be included in the design of a tour.

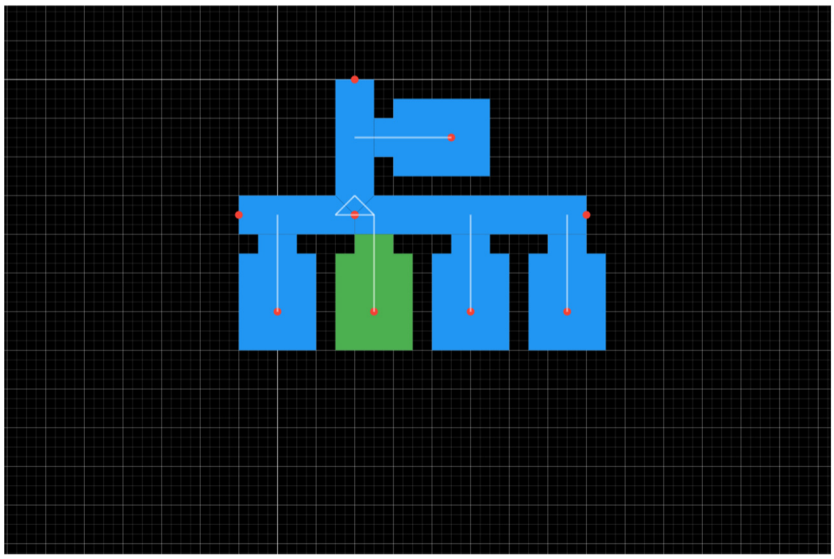

Figure 3. Recreation of an indoors space using the Cartographer tool.

 doi:10.20944/preprints202402.0404.v1

Another important step is the selection of the BLE beacons. The following two requirements exist for BLE beacons: The first is omnidirectional signal support and the second is iBeacon protocol support. Additionally, the high advertising frequency is important. The next step includes the actual placement of the beacons inside the room. The placement of the beacons does not have to be in the exact location defined by the generated map, but it is recommended to use a trial-and-error approach to find an optimal point that produces a uniform signal with the smallest possible standard deviation.

3.1.2. Physical and psychological user state assessment

Activity Monitoring (AM) is a well-established technique for quantitatively assessing the intensity of Physical Activity (PA) undertaken by an individual over time. AM has been used in various cases of obesity [20], Chronic Obstructive Pulmonary Disease – COPD and mental health [21]. It is often used alone, or alongside traditional methods based on questionnaires or interviews, to reduce inaccuracies, such as recall bias, inherent in these qualitative techniques [22], [23]. Devices used for AM often rely mainly on accelerometers, which measure the acceleration of the part of the body they are worn on.

One of the goals of the OPTORER project is to assess the physical state of the user, which is one of the criteria that can influence the touring, even dynamically. For this purpose, OPTORER includes a companion application for smartwatches with the Android Wear operating system. Its creation was necessary to obtain the user's biometric data, since the APIs of the Android Operating System for application communication with the user's smartwatch require that the applications running on the two devices have the same ID. However, the purpose and functionality of the user applications and services cannot be entirely based on it, as it is not necessary that all end users own a smartwatch. Therefore, in order not to restrict the use of the mobile phone application only to users who wear a smartwatch, the functionality of physical and emotional state assessment is optional, but available to the respective users, who will have to install the application for wearables to activate the feature.

More specifically, the assessment is based on the *MET (Metabolic Equivalent of Task)* measurement, which is a new measure developed to provide information on how people consume energy during the day. MET is a measure used in dietetics and expresses the metabolic load of an activity in relation to the basic metabolic rate [24]. This measure takes into account energy expenditure during hours of activity and during hours of rest, offering a complete, detailed map of a person's metabolic activity. Smartwatches and fitness trackers typically use their sensors (e.g., accelerometers, heart rate monitors) and other technologies, to record the user's activity and biometric data. They then analyze this data to produce MET metrics such as number of calories burned during the day, rest duration, activity times.

The interpretation of this data is done in the mobile application, based on the calculation of the MET of the user's walking and on his pace, [24]. It can be used to make qualitative inferences (if the user's activity is considered vigorous exercise) or, in combination with user details such as gender, weight and age, quantitative inferences (such as the user's caloric intake). By knowing the user's pace, energy consumption can be calculated followed the approach in [24]. By monitoring the result of this process, it is checked if the user has exceeded a threshold limit of 4 MET for the last 10 minutes, every 5 seconds. If so, the user receives a notification that gives them the option to take a shorter route if accepted.

As a result, the use of this measure allows the presentation of the user data from the tour and to personalize his/her navigation, based on the estimated physical condition, without requiring the input of personal information. Finally, in order to comply with the General Data Protection Regulation (GDPR), the data is processed entirely on the user's devices and the results are not sent to the backend.

3.1.3. Dynamic Multi-Criteria Routing Optimization

Problem Statement: The user starts an external tour from a geographic start point (START) with final destination a geographic arrival point (END), following a route that passes through a set of geographic points of interest (PoIs) and possibly a set of service points (STOPs). The START and END

 doi:10.20944/preprints202402.0404.v1

points can be the same (in case the user returns to his base after the tour) or different, while POIs or STOPs can be added by the user dynamically during the tour or even being removed. The sequence of visiting PoIs (SEQ - Sequence) can be defined from the beginning (in the case that this serves a thematic tour), or free (NOSEQ - No Sequence). The user, depending on the Means of Transport (s)he uses or not (MoT – Means of Transport) is properly routed to his final destination (END) through the points (s)he wants to visit (PoIs, STOPs). The routes suggested to the user are selected based on criteria which indicatively may be either the shortest in terms of distance, or the shortest in terms of time, or those that enhance the user's experience or allow better accessibility based on their abilities. Events (EVENTs) during a tour can dynamically reshape routing, making paths prohibitive in order to either protect the user, or avoid their discomfort, or facilitate the community.

One of the PoIs that a user visits may be an indoor space, such as a museum as long as the space is equipped with the BLE infrastructure described above. This allows the OPTORER mobile application to locate the user within it, with the user's starting point being an internal PoI (ISTART – Internal Start) and destination internally to another, or the same point (IEND – Internal End) passing through a series of internal PoIs (IPOIs – Internal POIs) and possible internal stops where services are provided (ISTOPs – Internal STOPs). Corresponding to the outdoor tour, the indoor tour can follow a visiting sequence of IPOIs (ISEQ – Internal Sequence) or not (INOSEQ – Internal No Sequence), while an internal event (IEVENT – Internal Event), or an external event (EVENT), can dynamically configure internal routing. For example, an indoor fire will lead the user to the emergency exit, but so will an external event such as an earthquake.

The tour may be subject to time restrictions, either as a whole (TTIME – Total Time – time from the moment of starting to the final destination) or regarding the visit to an individual POI named X (VTIME(X) – Visit Time to POI X – time of visit to point X).

Optimization Problem: The optimization problem that was addressed in OPTORER, for each user navigating externally, is to *find the optimal route based on the criteria and constraints to be set*, starting from the starting point (START) and ending at the arrival point (END) by traversing a series of POIs and STOPs (service points), which can be dynamically configured by the user by adding new ones or removing already declared ones.

The available routes from one point (START, POI, STOP) to another point (POI, STOP, END) arise depending on the MoTs used or not by the user, as well as other restrictions that may have to be satisfied. The best possible of the routes, taking into account in the solution of the problem the set of individual routes, is selected based on the criteria set and is expressed by an *objective mathematical function*, the value of which must be optimal (minimum or maximum) for the selected total path among all possible paths. Indicatively, the objective function can express the total distance to be traveled, or the total time of the route, or the cost of the route, or some other metric that expresses the user's experience during the route, or even weighted combination of more criteria than those mentioned and/or others.

Constraints of the problem may include the visiting sequence (SEQ, ISEQ) of PoIs to be already predetermined. This can serve the purposes of a thematic tour, inside a predefined tour that the user chooses, and which can be dynamically configured. Of course, the user can also deny the provided sequence for visiting PoIs. Visiting PoIs in a predetermined order reduces the complexity of the optimization problem, since it significantly limits the number of available routes considered when selecting the optimal one.

Modeling the optimization problem: Multiple optimization problems have been extensively studied in the past in various versions, which are suitable and can be modeled accordingly to form relating optimization problems like the one considered here. The benefit of such an approach is that it leverages years of research and development to fully understand the complexity of the problem, as well as the behavior and requirements of various solutions, with the goal of deriving an optimal or near-optimal solution in a short time for “large” problems, where the set of possible solutions to be considered is too many.

A well-studied routing problem, which can be generalized and properly formulated to express the optimization problem under consideration, is the **Vehicle Routing Problem (VRP)**. VRP is

 doi:10.20944/preprints202402.0404.v1

defined as that set of problems in which fleets of trucks are deployed in one or more depots. These fleets are required to serve a number of customers in pre-defined geographic locations. *The goal of VRP is to determine the appropriate routes to serve customers, with the minimum cost* [25]. In classic VRP, each vehicle starts from a depot and returns to the same depot after uniquely visiting a set of customers assigned to it on its decided route. A wide range of variations of the problem have been studied in the literature. When considering one depot and one vehicle, *only then the problem is the same as the well-known TSP (Travelling Salesman Problem)* [26], which was essentially the primary problem studied and from which VRP arose.

VRP belongs to the class of those problems where the decision of the optimal solution is NP-hard [27], since there is no algorithm of polynomial complexity to solve the problem. This translates that VRP, as well as the problem addressed in OPTORER, belongs to a class of problems where the size of the problems (equivalent input size, e.g. number of customers, trucks, depots in the VRP) that can be solved by mathematical methods, or combinatorial optimization, is limited. This is because of the computing power, and thus the time required to find an optimal solution, as the problem size grows becomes prohibitive (i.e., it exponentially increases with problem size). Therefore, finding a globally optimal solution is time-realistic only for small-scale problems. However, a series of heuristics and metaheuristics algorithms that have been developed aim to find a good enough solution close to the optimal solution (i.e., near optimal solution) if not the optimal solution, with an intelligent and non-exhaustive search for solutions in short times and for large problem sizes.

A *heuristic algorithm* is essentially a technique designed to solve a problem faster, while a *meta-heuristic* is a higher-level process or heuristic designed to find, generate, tune, or select a heuristic that can provide a reasonably good solution to an optimization problem. Popular heuristic/meta-heuristic algorithms are Tabu Search, Simulated Annealing and Genetic [27]. Variations of the VRP problem arise by considering different constraints taken into account and involving information on the characteristics of customers, vehicles, routes, delivered products, and drivers, as well as by considering different optimization objectives expressed by the appropriate objective functions.

The *mathematical formulation* of the classical Capacitated Vehicle Routing Problem (CVRP) concerns a fleet of K identical vehicles (of the same capacity C) making deliveries to customers from a central depot. CVRP can be described as the following graph theoretic problem. Let G=(V,A) be a complete graph, where V={0,1,…,n} is the set of vertices and A is the set of edges. Nodes i=1,…,n correspond to the customers, while node 0 corresponds to the depot. A non-negative cost, c_{ij}, is associated with each arc (i,j)∈A and represents the travel cost spent to go from node i to node j. We consider a directed graph, therefore $c_{ij} \neq c_{ji}$.

Each customer i (i=1,…,n) is associated with a known non-negative demand, d_i, to be delivered and the depot has a fictitious demand $d_0=0$. Given a set of nodes S ⊆V, let $d(S) = \sum_{i \in S} d_i$ denote the total demand of the subset of customers. From the vehicle routing problem, K vehicle routes should be generated, where each route will start and end at the depot and each vehicle will visit a subset of customers. Each vehicle will perform only one route, while all vehicles have the same capacity. Each customer must be visited by only one vehicle, while the total demands of customers visiting a vehicle on a route must not exceed its capacity. The specific routes will be chosen so as to minimize distribution costs.

Therefore, from the above results the mathematical formulation of the problem, as presented as an extension of the TSP problem formulation by Dantzig, Fulkerson and Johnson to create a two-index vehicle flow formulation in VRP:

$$min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

subject to

$$\sum_{i \in V} x_{ij} = 1 \quad \forall j \in V \setminus \{0\} \tag{1.1}$$

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i \in V \setminus \{0\} \tag{1.2}$$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 February 2024 doi:10.20944/preprints202402.0404.v1

$$\sum_{i \in V \setminus \{0\}} x_{i0} = K \tag{1.3}$$

$$\sum_{j \in V \setminus \{0\}} x_{0j} = K \tag{1.4}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset \tag{1.5}$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in V \tag{1.6}$$

In this formulation c_{ij} represents the cost of going from node i to node j, x_{ij} is a binary variable that has a value of 1 if the arc going from i to j is considered part of the solution and a value of 0 otherwise, K is the number of available vehicles and r(S) corresponds to the minimum number of vehicles required to serve (in terms of its demands) the set S. We also assume that 0 is the depot node.

Constraints 1.1 and 1.2 state that exactly one arc enters and exactly one exits each vertex associated with a customer, respectively. Constraints 1.3 and 1.4 say that the number of vehicles leaving the depot is the same as the number entering. Constraints 1.5 are Capacity-Cut Constraints (CCCs), which enforce that the routes must be connected and that the demand on each route must not exceed the vehicle's capacity. The capacity cut constraints remain correct if r(S) is replaced by the lower bound $\lceil d(S)/C \rceil$ where C is the capacity of each vehicle and d(S) is the total customer demands. Finally, constraints 1.6 are the integrity constraints. In the formulation it has been assumed that the demands of any customer do not exceed the capacity of the vehicle.

The mathematical formulation of the problem has been used extensively to model the basic VRP (CVRP) and VRPB (VRP with Backhauls is an extension that includes both a set of customers to whom products are to be delivered, and a set of vendors whose goods need to be transported back to the distribution center). However, it is considered suitable for these simple problems. It can be used only when the cost of the solution can be expressed as the sum of the costs of the arcs. We also cannot know which vehicle crosses each arc. Therefore, we cannot use it for more complex models, where the cost or even the feasibility of the solution depends on the customer order, or the vehicles used.

We appropriately formulate the mathematical formulation of the classical VRP problem as we presented it, in order to express the case where we have a single vehicle that has the ability to meet the demands of all n customers, while it starts its route from a warehouse 0 and necessarily ends at a second warehouse n+1 optimally visiting all n customers/locations. Essentially in this version of the problem we are describing, the problem is that of the TSP traveling salesman, with the variation that he does not return to the point of departure but terminates the route at some other given point.

Figure 4 graphically illustrates the solution to the VRP variant we described without specifying the values of the problem parameters.

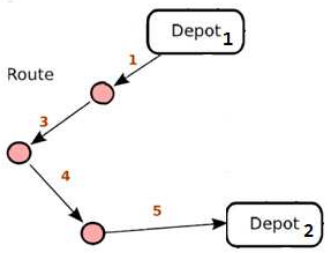

Figure 4. Illustration of the solution of the variant of the classical VRP problem with one vehicle and different arrival warehouse.

We set the number of available vehicles K=1, the minimum number of vehicles to serve customers' demands $\boldsymbol{r(S) = 1}$, the 2nd depot where the vehicle must end its route as depot n+1 and modify condition 1.2 to express that the vehicle will terminate at depot n+1 and not at depot 0 from which it departed.

 doi:10.20944/preprints202402.0404.v1

This results in the following mathematical formulation expressing the variant of the classic VRP we described:

$$min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

subject to

$$\sum_{i \in V} x_{ij} = 1 \quad \forall j \in V \setminus \{0, n+1\} \tag{2.1}$$

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i \in V \setminus \{0, n+1\} \tag{2.2}$$

$$\sum_{i \in V \setminus \{0,n+1\}} x_{in} = 1 \tag{2.3}$$

$$\sum_{j \in V \setminus \{0,n+1\}} x_{0j} = 1 \tag{2.4}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq 1, \quad \forall S \subseteq V \setminus \{0, n+1\}, S \neq \emptyset \tag{2.5}$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in V \tag{2.6}$$

This variation of the classic VRP problem finds an exact correspondence with the optimization problem we are asked to address in OPTORER if we consider the departure depot as the START point (or ISTART internally), the route ending depot as the END point ((or IEND indoors) and customer locations as POIs or STOPs ((or IPOIs or ISTOPS internally). Figure 5 graphically illustrates the solution to an OPTORER service routing problem.

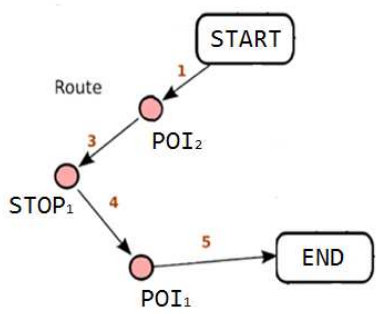

Figure 6. Illustration of the solution of the variant of the classic VRP problem with one vehicle and a different arrival depot with a direct mapping to the optimization problem we are asked to solve in the OPTORER service.

In the case that the sequence of visits is initially predetermined in whole or in part (e.g., the visit to POIx must be preceded by a visit to POIy), then the solution in whole or in part is known to the static problem. Thus, part of the solution as expressed by the parameters $\boldsymbol{x_{ij} \in \{0,1\} \quad \forall i,j \in V}$ is known and is introduced as a constraint in the problem formulation by setting the parameter $\boldsymbol{x_{ij} = 1}$ if the visit to node i must necessarily be followed by visiting node j.

Input to the problem should be the costs of the individual paths between all the nodes of the graph. Deriving the cost of a route from one node to another node depends on what it represents, as well as whether or not the means of transportation are used. Illustratively, the cost may express the distance (depending on the MoT) between the nodes and/or the travel time (depending on traffic or other factors) and/or the user experience (based on an assessment of inconvenience or pleasure in a nice ride).

The derivation of the cost and the selection of the individual path from one point to another is also a subproblem that must be solved before the optimization problem we are considering. Algorithms commonly used by services such as Google Maps to choose a suitable route on the map

 doi:10.20944/preprints202402.0404.v1

and based on other factors such as traffic are Dijkstra's [29] and A* [30] for finding the minimum path in a graph where the edges have weights.

The navigation of the user from one point to another on the path that has resulted from solving the optimization problem is a different function of the OPTORER system, however the navigation should be done on the individual path that has been chosen based on cost.

In OPTORER service, the problem is not considered as static but as *dynamic*, where visiting points can be changed along the route according to the traveler's desire, the MoT (e.g., a part of the route can be done by car and then the browser continues on foot), some event triggers a change to the route or changes the costs of specific routes, or the tour from an outdoor area is continued in an indoor area that supports the technology introduced by OPTORER. The obvious thing is that any change triggers the system to re-solve the problem with the entry of the new data. However, given the complexity of the problem, techniques to partially solve it are considered with the goal of near-real-time decision making.

Considering the OPTORER service optimization problem as a variant of the VRP problem, rather than a variant of the TSP problem, in the first place is a tactical choice, as future extensions of the service will be able to rely on an extension of the modeling and implementation to be done. Possible extensions include adopting time windows, supporting mass transit of travelers, controlling the attendance of travelers using the service at points of interest such as museums based on capacity, etc..

Optimization Engine Implementation: OptaPlanner [31] is the choice for implementing the optimization engine of the OPTORER service. Apart from being a free software solution, OptaPlanner also provides excellent possibilities for solving optimization problems with heuristic algorithms (such as tabu search, simulated annealing), while enabling the solution to be easily integrated with a number of other modern technologies required to implement an optimization engine in the context of a high-scale service (e.g., openshift, quarkus, spring boot), while accepting input and providing output via APIs REST APIs in JSON format, which further facilitates its integration as a module of a system like OPTORER. In addition, it supports a number of techniques that facilitate solving dynamic problems in near real time (e.g., real time planning).

The process of solving a problem with OptaPlanner consists of modeling the problem as a set of classes, appropriately annotated based on their role (PlanningSolution, PlanningEntity etc.), defining constraints (as mandatory - Hard and flexible – Soft), the definition of the function to be optimized (score function as it is called), the selection and configuration of the algorithm that will solve the problem, the loading of the input data to the problem and finally the execution of its solution.

OptaPlanner integrates well, as demonstrated in various demo applications, with open-source routing engines such as GraphHopper [32] (but also Google Maps), which is a map-optimized route engine using OpenStreetMap mapping [33], which provides open mapping data. GraphHopper uses efficient algorithms to decide the route on a road network, such as Dijkstra and A* algorithm, and it can be configured and modified appropriately so that the individual route decision is according to the intended goals. Also, excellent integration has been demonstrated for visualization of results using Leaflet.js [34] and also Google Maps. For our solution we adopt Timefold [35] which is a latest different path to the implementation of Optaplanner with exceptional refined characteristics while we integrate it with Graphhopper.

Figure 7 illustrates an indicative tour on the map as a visual representation of the solution to an optimization problem we consider in OPTORER; the Start point is the first on the right on the map, the Arrival point is the last on the left on the map, the intermediate points are visited in a sequential manner as defined by the numbers that signify them).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 February 2024 doi:10.20944/preprints202402.0404.v1

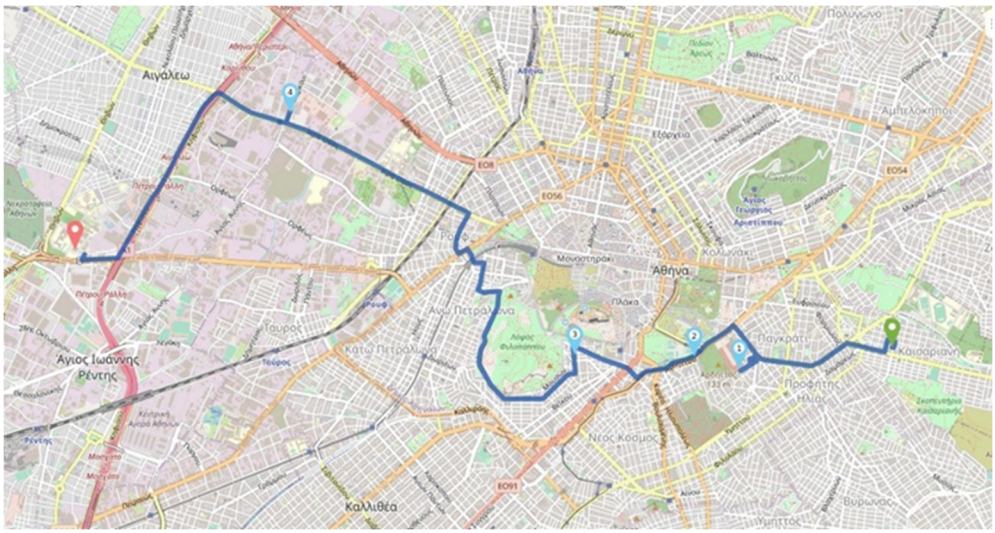

Figure 7. Indicative tour and road route.

4. Evaluation Results

For evaluation purposes, tests for the main components of the OPTORER service were conducted. Therefore, evaluation took place for the Mobile Application and the Routing Optimization process. It should be noted that evaluation tests for the indoor localization and positioning service were conducted and can be found in [19]. The results of the evaluation process are:

- *Mobile Application Evaluation Results:* During the test, there were certain KPIs that needed to be met in order to make sure that the performance of the modules was successful. Table 1 presents the KPIs that were tested during the evaluation of the mobile application along with the results from the measurements that were captured:

Table 1. Tested KPIs and Results for Module Performance.

KPI	Target Value	Achieved Value	Tool Used
User Location Time	< 3 sec (on avg)	< 65 msec	Flutter Dart Code
User Location Accuracy	< 1.5 m	~1 m	Flutter Dart Code
Time to load the Application	< 3 sec	< 1.58 sec	Firebase APM
Memory Usage in mobile phone during app execution	<= 512 MB	<150 MB	Flutter DevTools
Data / min needed to follow a route in the mobile phone (on avg)	<= 1 MB / min [36]	< 500 KB / min	New Relic APM360
Time to place the user location on the map	<= 20 sec [37] [38]	< 4 sec	New Relic APM360
Average battery consumption on the mobile phone	<= 150 mAh [39]	< 90 mAh	AccuBattery App
Time to load the map (Example: when moving from outdoors to indoors)	<= 10 sec	< 3 sec	Flutter Dart Code

For the Mobile Application, the related indicators were measured using either separate blocks of code running simultaneously with the application code or using one of three separate tools. These include the Firebase Performance Monitoring tool [40], offered by Google, the New Relic Performance Monitoring Tool [41] and the Android Studio [42] integrated performance measuring tool which is called Flutter DevTools [43]. Lastly, due to the fact that battery monitoring requires very specific and native permissions given from the device, the battery consumption of the application was measured

 doi:10.20944/preprints202402.0404.v1

using the AccuBattery Application [44] during in-lab testing. A demo route (approximately 10 min in duration) was designed and carried through, including rerouting due to path drifting, during which the AccuBattery Application was activated on the same device.

The monitoring process of these tools has produced a number of charts showing the above achieved values, regarding the mobile application memory and network performance. It should be noticed that for the KPIs referring to the start time of application as well as the average battery consumption, the device's hardware capabilities could significantly affect the performance, presenting high fluctuations. The performance monitoring charts are shown in Appendix A.

- *"Route Engine" Evaluation Results:* The solution of the routing problem is activated by a call providing the appropriate input data, to the complex module that we call "Route Engine". Calculation of an optimal route is initiated while upon completion of the solution, the output data describing the route to be followed by the user, is returned. The route is initially calculated before the start of the tour and re-calculated whenever the tour or the conditions in which it takes place change.

The "Route Engine" consists of the OPTPROXY proxy service, the GraphHopper routing engine and the Timefold Optimization Engine. The individual engines internally complete additional modules such as the OpenStreetmap implementation and the inference and rules engine. A call to find an optimal path of a tour is made to the OPTPROXY proxy service where the request specifies whether a sequence of points defining the tour also matches the visit sequence during the tour or not. In the first case the proxy application will send a query to the GraphHopper engine and return the response it will receive through OPTPROXY, while in the second case it will first send an appropriately formatted query to the Timefold engine and after receiving the response with the optimal visit order will appropriately configure the original request and will forward it to the GraphHopper engine from where it will receive the response that will be returned via OPTPROXY. The Timefold engine is only queried if the visit points are not given in their desired visiting order, and thus the order that minimizes the total distance traveled or total time spent to complete the tour must be found. It is clear that the significant time will be spent in the optimization engine solving the computationally complex problem of finding the optimal visiting order whenever this is required.

In the following performance study, we conduct a worst-case analysis assuming that all requests refer to tours with a free order of visit while each request is for the maximum number of visit points that we consider a tour may include. In the performance study we consider and study the instantaneous number of concurrent requests to find optimal routing. Such a worst-case study approach will lead us to a safe dimensioning, parameterization and forecasting of the resources we must have available to continuously achieve our goals. In addition, the quality of the solution will be evaluated, in the form of deviation from the optimal one, in relation to the resources and time available for finding it.

In the following scenario, the correlations between load expressed in number of simultaneous requests, computing resources, quality of generated solution and request processing time are studied. The results of the study lead to the optimal parameterization of the system and the individual routing and optimization engines so that within the targeted times and the targeted maximum number of requests, the best possible result is produced utilizing the available computing resources.

The OPTPROXY application, the GraphHopper engine (with Attica's roadmap) and the Timefold engine run on the same virtual machine provided by a rather stable infrastructure exhibiting very low variation in the share of physical resources along time. We use a virtual machine with a processor (CPU) with 8 cores (cores) and 16GB memory. In order to have a view of the available resources we run the PassMark [45] benchmark. By studying the variability of resources over time by running the benchmark at different times, we conclude that the variability is low enough which confirms the stability of resources over time. The PassMark CPU mark metric reflecting the computational power of the virtual machine has a value close to 8000 which ranks the virtual machine's CPU computational performance in the mid- to high-performance CPUs [46]. The GraphHopper engine runs using the Attica road map. The Timefold machine is configured to give up to 5 seconds to solve each optimization problem, while stopping the solution if in 1 second it has not found a better solution than the previous solution it had found to the problem. The number of

 doi:10.20944/preprints202402.0404.v1

problems allowed to be solved by the engine at the same time (number of simultaneous solvers – solvers) is studied by setting as a value 8 (1 thread per core), 16 (2 threads per core), 32 (4 threads per core), 64 (8 threads per core) and the 96 (12 threads per core). We use the same request describing a tour each time, which consists of a total of 10 points including the starting and destination points without a predetermined order of visit. We note that the problem is solved anew each time for each new request, while the selection of points consisting the tour has been made so that finding the route is classified as complex. Given the nature of our service, each tour requested by a user is expected to include from 2 to a maximum of 10 visit points including the starting point and the destination. The trial tour is shown on the map in Figure 8 (start point first on the right on the map, arrival point last on the left on the map, intermediate visit points in a sequential manner as defined by the numbers that mark them). The route shown is the visualization of the solution to the routing problem.

Figure 8. Trial tour used in our evaluation and visualization of the solution to the routing problem as a road route.

We run each experiment once and present the results, considering that the variation we will have in the value of the considered metrics will be small in repeated experiments due to the exclusive use of the virtual server as well as the observed constancy of infrastructure resource performance. This claim was confirmed by multiple runs of the same experiment. The following table lists the minimum, average, and maximum request processing time achieved for different values of concurrent requests and active concurrent solver threads. The quality of the solution we achieve in all the cases we consider is the optimal one.

Table 2. Results for the minimum, average, and maximum request processing time achieved for different values of concurrent requests and active concurrent solver threads.

Concurrent requests	Active solvers	Minimum request processing time (in seconds)	Average request processing time (in seconds)	Maximum request processing time (in seconds)
8	8	1,25	1,28	1,3
16	8	1,28	1,76	2,23
32	8	1,07	2,44	3,73
64	8	1,32	4,51	7,79
128	8	1,32	6,86	14,69
200	8	1,32	9,53	21,42
8	16	1,27	1,29	1,32
16	16	1,27	1,33	1,4
32	16	1,16	1,76	2,36

 doi:10.20944/preprints202402.0404.v1

64	16	1,24	2,42	3,57
128	16	2,21	4,85	8,51
200	16	2,56	6,54	11,16
8	32	1,27	1,28	1,29
16	32	1,27	1,31	1,37
32	32	1,29	1,50	1,93
64	32	1,12	1,82	2,48
128	32	2,31	3,27	4,87
200	32	2,36	4,55	7,44
8	64	1,28	1,29	1,3
16	64	1,28	1,30	1,33
32	64	1,18	1,52	2,25
64	64	1,1	1,81	2,5
128	64	1,38	2,67	3,63
200	64	1,72	3,55	5,76
8	96	1,25	1,27	1,29
16	96	1,24	1,26	1,3
32	96	1,25	1,39	1,55
64	96	1,19	1,91	2,74
128	96	1,77	2,54	3,28
200	96	1,37	3,60	6,53

We observe that having 32 to 96 concurrent solvers provide for an average request processing time below 5 seconds even in the case of 200 concurrent requests. It is clear that as the number of concurrently active solvers increases the computational power available for a single request decreases. However, when concurrent requests exceed the number of concurrently active solvers, the requests not being able to receive service on arrival will be queued waiting for service when resources will be available. So, there is a trade off between computational power a single request is able to receive vs queueing delay it may suffer in order to receive that computational power. One could think that the best decision would be to have as much active solvers as the number of CPU cores available in order each optimization problem to get a good or optimal solution in less time. This is true when we consider quite larger instances of the problem (e.g. a tour with 200 points considered) requiring significantly more time for the given computational resources to be solved with a good solution provided. It is then that the queueing delay becomes less significant since the real tradeoff is between computational power, time and quality of the solution. In our case, given the computational resources available, the target times to provide a solution, the expected number of points a tour includes and the maximum number of concurrent requests considered, we achieve an optimal solution to the problem at all times and our concern is to find the number of active solvers and maximum number of concurrent requests a "route engine" is permitted to serve in order to provide the quality of service targeted. The scaling of the "Route Engine" complex module in the current version of OPTORER is achieved by sharing the load among several virtual servers hosting a "Route Engine" in a round robin or weighted according to their computational power fashion.

5. Conclusions

This paper presents the results for the design an implementation of a novel routing and touring service initially scheduled for the Attica region in Greece. The work was conducted for the OPTORER project and in order to be delivered certain technological challenges needed to be addressed. Indoor Routing, Physical Assessment and Dynamic Multicriteria Outdoors routing were the three main

 doi:10.20944/preprints202402.0404.v1

challenges, and a description of the implemented solutions was presented here. The results of the main modules evaluation were, also, presented and discussed.

There also some additional features that OPTORER service possesses that were not discussed in detail, but they add value to the overall user experience like the tours marketplace, allowing the end-users to select preselected tours that were designed and offered for everyone to buy. Additionally, the possibility for third party service providers to add their service inside the system so that it can be selected either from a tour organizer in a pre-selected tour or from the system when it dynamically responds to a change in the conditions (either physical condition due to fatigue or environmental conditions due to an accident). Finally, the reporting of incidents from the users and the assessment of the tours can help the system prioritize the available tours and act when necessary to provide alternate routes to the users that are affected by any dynamically changes to their tour.

Finally, for the future, possible extensions to the service may include the use of Artificial Intelligence (AI) to improve the physical assessment and the interconnection of the service with other services to receive alerts about the weather or emergency notifications that could demand an alert to be provided to the affected users.

Author Contributions: "Conceptualization, D.K, C.V. and H.L.; methodology, D.K.; software, M.P, D.M, C.V.; validation, M.P. and C.V.; formal analysis, D.G and C.V.; investigation, M.P and C.V.; resources, D.K.; data curation, M.P and C.V.; writing—original draft preparation, D.K, M.P and C.V; writing—review and editing, D.P and H.L.; visualization, D.P and C.V.; supervision, H.L.; project administration, H.L.; funding acquisition, C.V, D.K and H.L. All authors have read and agreed to the published version of the manuscript."

Funding: This research was funded by the European Regional Development Fund (ERDF) within the Regional Operational Program "Attica" under the call "Research and Innovation Synergies in the Region of Attica" (Project code: ATTP4-0353772).

Data Availability Statement: The data presented in this study are available on request from the corresponding author after the completion of the research project.

Conflicts of Interest: "The authors declare no conflicts of interest."

Appendix A : Optorer Mobile Application Memory and Network Performance

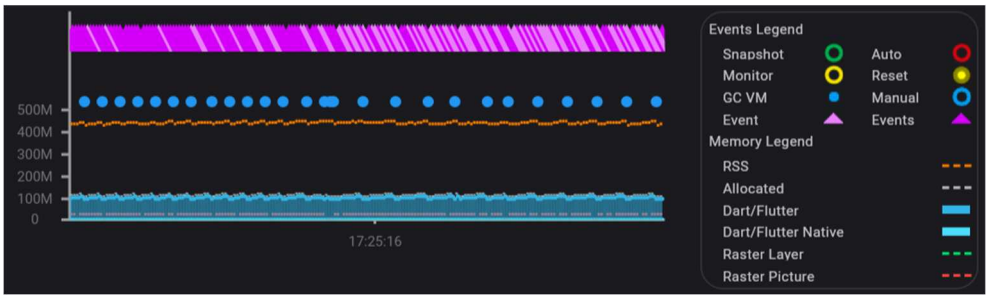

Figure 9. Android Studio DevTools: Optorer Application Memory Performance (Bottom Blue Graph – Dart / Flutter).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 February 2024 doi:10.20944/preprints202402.0404.v1

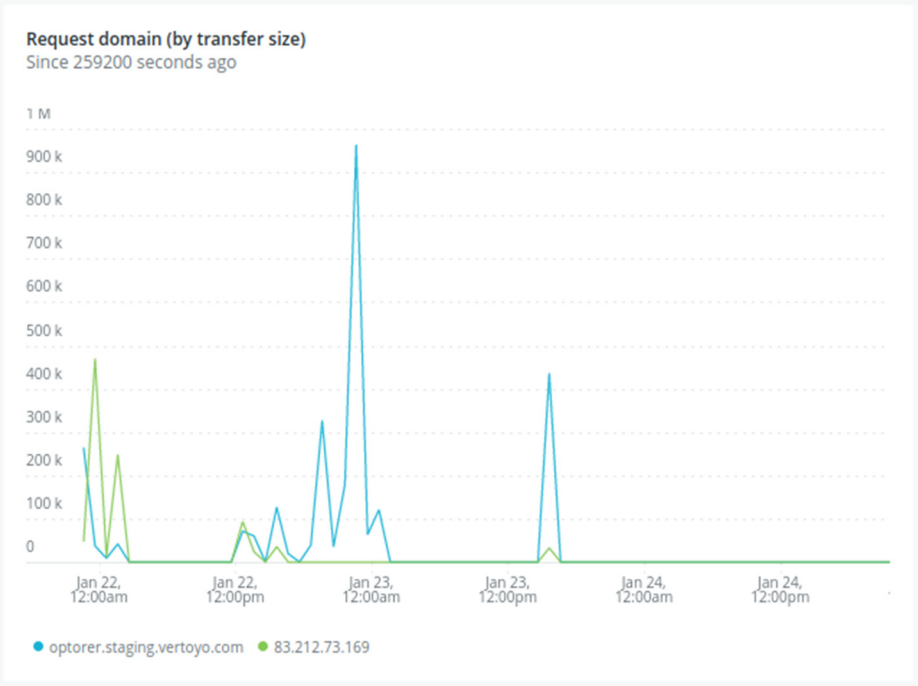

Figure 10. Average HTTP Transfer Size during navigation (green graph).

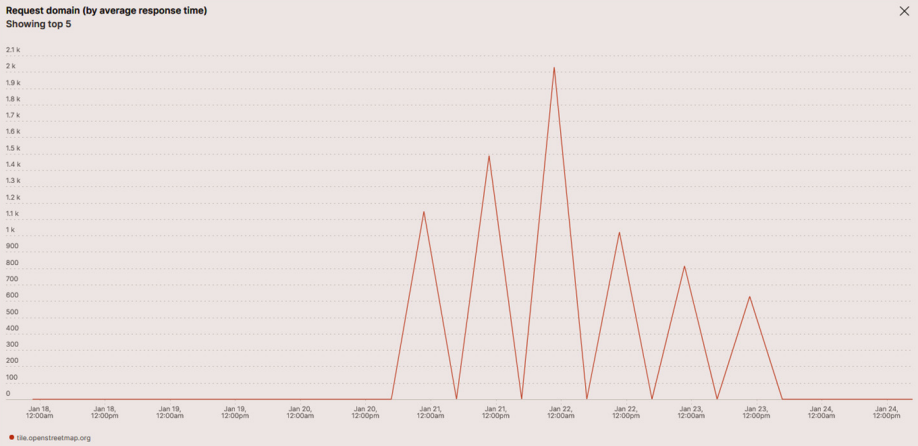

Figure 11. Average HTTP Response time during navigation.

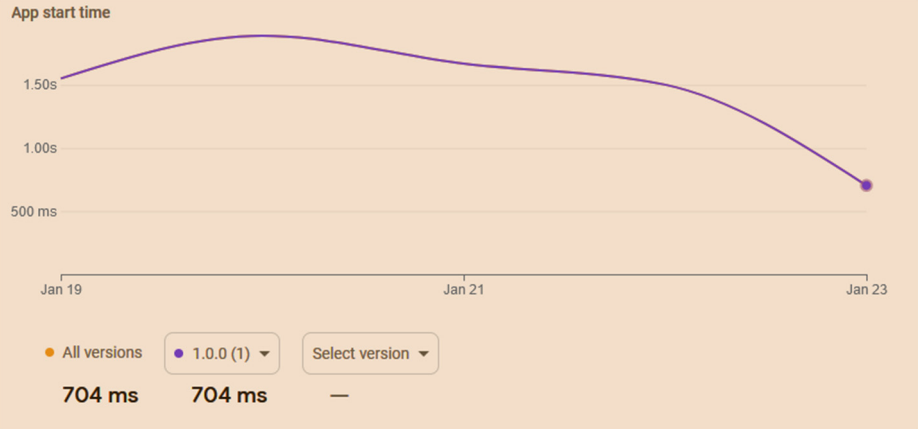

Figure 12. Average App Start Time.

 doi:10.20944/preprints202402.0404.v1

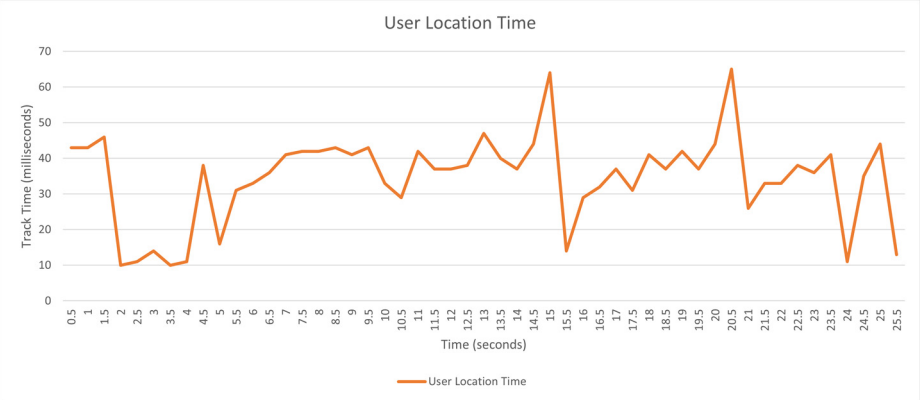

Figure 13. User Location Time.

References

1. A. Kontogianni and E. Alepis, "Smart tourism: State of the art and Literature Review for the last six years," *Array, vol6,* Jul. 2020.
2. F. Mehraliyev, I. C. Chan, Y. Choi, M. A. Koseoglu, and R. Law, "A state-of-the-art review of smart tourism research," *Journal of Travel and Tourism Marketing,* vol. 37, no. 1, pp. 78-91, 2020,.
3. J. Dorcic, J. Komsic, and S. Markovic, "Mobile Technologies and applications towards Smart Tourism – State of the art," *Tourism Review,* vol. 74, no. 1, pp. 82-103, 2019.
4. OPTORER project, Optimal routing and exploration of touristic and cultural areas of interest within Attica given personalized adaptive preferences, promoted underlying purpose and interactive experience - OPTORER PPE: ATTP4-0353772, online link: https://www.optorer-project.gr/.
5. Bekkelien, Anja, Michel Deriaz, and Stéphane Marchand-Maillet., "Bluetooth indoor positioning," Master's thesis, University of Geneva (2012).
6. Christoph Fuchs, Nils Aschenbruck, Peter Martini, and Monika Wieneke, "Indoor tracking for mission critical scenarios: A survey," *Pervasive and Mobile Computing,* vol. 7(1), pp. 1-15, 2011.
7. Guolin Sun, Jie Chen, Wei Guo, K.J. Ray Liu, "Signal processing techniques in network-aided positioning: a survey of state-of-the-art positioning designs," *IEEE Signal Processing Magazine,* vol. 22, no. 4, 2005.
8. Mayu Sumida, Teruhiro Mizumoto, Keiichi Yasumoto, "Estimating heart rate variation during walking with smartphone," in *UbiComp '13: Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing, Pages 245*, September 2013.
9. Paolo Toth and Daniele Vigo (Eds.), *The vehicle routing problem,* USA: Society for Industrial and Applied Mathematics, 2021.
10. "Optaplanner metaheuristics engine," [Online]. Available: https://www.optaplanner.org/.
11. "GraphHopper," [Online]. Available: https://www.GraphHopper.com/.
12. A. Riady and G. P. Kusuma, "Indoor positioning system using hybrid method of fingerprinting and pedestrian dead reckoning," *Journal of King Saud University - Computer and Information Sciences,* vol. 34, no. 9, p. 7101–7110, 2022.
13. N. Pakanon, M. Chamchoy and P. Supanakoon, "Study on accuracy of trilateration method for indoor positioning with Ble Beacons," in *6th International Conference on Engineering, Applied Sciences and Technology (ICEAST) [Preprint], Available at: https://doi.org/10.1109/iceast50382.2020.9165464.* , 2020.
14. T. Kluge, C. Groba and T. Springer, "Trilateration, Fingerprinting, and Centroid: Taking Indoor Positioning with Bluetooth LE to the Wild," in *IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM),* Cork, Ireland, 2020, pp. 264-272, doi: 10.1109/WoWMoM49955.2020.00054.
15. F. Zafari and J. Gao, "A survey of indoor positioning systems and technologies," *Journal of Location Based Services,* vol. 10, no. 3, pp. 180-206, doi: 10.1080/17489725.2016.1202965, 2016.
16. N. S. Kodippili and D. Dias, "Integration of fingerprinting and trilateration techniques for improved indoor localization," in *Seventh International Conference on Wireless and Optical Communications Networks - (WOCN),* Colombo, Sri Lanka, 2010, pp. 1-6, doi: 10.1109/WOCN.2010.5587342.
17. A. Sashida, D. P. Moussa, M. Nakamura and H. Kinjo, "A Machine Learning Approach to Indoor Positioning for Mobile Targets using BLE Signals," in *34th International Technical Conference on*

Circuits/Systems, Computers and Communications (ITC-CSCC), JeJu, Korea (South), 2019, pp. 1-4, doi: 10.1109/ITC-CSCC.2019.8793423.

18. P. Sthapit, H.-S. Gang and J.-Y. Pyun, "Bluetooth Based Indoor Positioning Using Machine Learning Algorithms," in *IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia)*, JeJu, Korea (South), 2018, pp. 206-212, doi: 10.1109/ICCE-ASIA.2018.8552138.
19. Margaritis, D., Leligou, H.C., Kogias, D.G., "One Dimensional Fingerprinting as an Alternative to the Free Space Path Loss Equation for Indoor Positioning". In: Novel & Intelligent Digital Systems: Proceedings of the 3rd International Conference (NiDS 2023). NiDS 2023. Lecture Notes in Networks and Systems, vol 783. Springer, Cham. https://doi.org/10.1007/978-3-031-44097-7_32.
20. A. . G. Bonomi and K. R. Westerterp, "Advances in physical activity monitoring and lifestyle interventions in obesity: A review," *Int. J. Obesity*, vol. 36, pp. 167-77, Feb.2012.
21. C. A. Janney, A. Fagiolini, H. A. Swartz, J. M. Jakicic, R. G. Holleman and C. R. Richardson, "Are adults with bipolar disorder active? Objectively measured physical activity and sedentary behavior using accelerometry," *J. Affective Disorders*, Vols. 152-154, pp. 498-504, Jan. 2014.
22. J. J. Reilly, V. Penpraze, J. Hislop, G. Davies, S. Grant and J. Y. Paton, "Objective measurement of physical activity and sedentary behaviour: Review with new data," *Arch. Dis. Childhood*, vol. 93, pp. 614-619, Jul. 1, 2008.
23. L. M. Reiser and E. A. Schlenk, "Clinical use of physical activity measures," *J. Amer. Acad. Nurse Practitioners*, vol. 21, pp. 87-94, 2009.
24. M. Abel, J. Hannon, D. Mullineaux and A. Beighle, "Determination of step rate thresholds corresponding to physical activity intensity classifications in adults.," *J Phys Act Health*, vol. 8, no. 1, pp. 45-51, doi: 10.1123/jpah.8.1.45. PMID: 21297184., 2011 Jan.
25. P. Toth and D. Vigo, *An overview of vehicle routing problems, in The Vehicle Routing Problem (P. Toth and D. Vigo eds.)*, SIAM Monographs on Discrete Mathematics and Applications, 2002.
26. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley & Sons, ISBN 0-471-90413-9., 1985.
27. D. P. Bovet and P. Crescenzi, *Introduction to the Theory of Complexity. Prentice Hall. p. 69*, ISBN 0-13-915380-2., 1994.
28. L. Bianchi, M. Dorigo, L. M. Gambardella and W. J. Gutjahr, *A survey on metaheuristics for stochastic combinatorial optimization*, Natural Computing. 8 (2): 239–287., 2009.
29. E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269-271, 1959, doi: 10.1007/BF01386390. S2CID 123284777.
30. N. J. Nilsson, *The Quest for Artificial Intelligence*, Cambridge: Cambridge University Press. ISBN 9780521122931., 2009-10-30.
31. "OptaPlanner," [Online]. Available: https://www.optaplanner.org/.
32. "GraphHopper," [Online]. Available: https://www.graphhopper.com/.
33. "OpenStreetMap," [Online]. Available: https://www.openstreetmap.org/about.
34. "Leaflet.js," [Online]. Available: https://leafletjs.com/.
35. "Timfold", [Online]. Available: https://timefold.ai.
36. eSim Europe, "How much data does Google Maps use?", March 2023, Web: https://europeesim.com/blog/how-much-data-does-google-maps-use/
37. López, J.M.L., Aguilar, F.L., Abascal, J.J.C. (2012). A-GPS Performance in Urban Areas. In: Rodriguez, J., Tafazolli, R., Verikoukis, C. (eds) Mobile Multimedia Communications. MobiMedia 2010. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 77. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35155-6_33
38. Frank Van Diggelen, "Google to improve urban GPS accuracy for apps", December 2020, Web: https://www.gpsworld.com/google-to-improve-urban-gps-accuracy-for-apps/
39. Greenspector & ATOS, "Consumption of TOP 30 most popular mobile applications", July 2019, Web: https://greenspector.com/wp-content/uploads/2020/01/Atos-GREENSPECTOR-TOP30-benchmark-english.pdf.
40. "Firebase Performance Monitoring", [Online], Available: https://firebase.google.com/products/performance
41. "New Relic APM 360", [Online], Available: https://newrelic.com/platform/application-monitoring
42. "Android Studio IDE", [Online], Available: https://developer.android.com/studio

 doi:10.20944/preprints202402.0404.v1

43. “Flutter DevTools”, [Online], Available: https://docs.flutter.dev/tools/devtools/overview
44. “AccuBattery App [Online], Available: https://accubatteryapp.com/
45. “Passmark”, [Onkline], Available: https://www.passmark.com/
46. “PassMark - CPU Mark, High Mid Range CPUs” , [Online], Available: https://www.cpubenchmark.net/mid_range_cpus. html.