Vulcanus 2.0: A Recommender System for Accessibility

Ismael Gomes Cardoso, Bruno Mota,
Jorge Luis Victória Barbosa, Rodrigo da Rosa Righi
University of Vale do Rio dos Sinos, Applied Computing Graduate Program,
Av. Unisinos, 950 - São Leopoldo, RS, Brazil,
ismaelcardoso@yahoo.com.br, brunomota55@gmail.com,
jbarbosa@unisinos.br, rrrighi@unisinos.br

Abstract

Even though the use of recommender systems is already widely spread in several application areas, there is still a lack of studies for accessibility research field. One of these attempts to use recommender system benefits for accessibility needs is Vulcanus. The Vulcanus recommender system uses similarity analysis to compare user's trails. In this way, it is possible to take advantage of the user's past behavior and distribute personalized content and services. The Vulcanus combined concepts from ubiquitous computing, such as user profiles, context awareness, trails management, and similarity analysis. It uses two different approaches for trails similarity analysis: resources patterns and categories patterns. In this work we performed an asymptotic analysis, identifying Vulcanus' algorithm complexity. Furthermore we also propose improvements achieved by dynamic programming technique, so the ordinary case is improved by using a bottom-up approach. With that approach, many unnecessary comparisons can be skipped and now Vulcanus 2.0 is presented with improvements in its average case scenario.

Keywords: Recommender System, Accessibility, Context Awareness, Asymptotic Analysis.

1 Introduction

Technology is in the era of big data [1], where information is abundant and even generated in excess. People make use of a variety of strategies in order to take daily decisions on what to buy, on how to make better use of their free time and even on whom to date. Recommender Systems (RSs) arise as a way to automate some of these strategies, intending to provide accessible and customized recommendations with high quality and precision [2].

Both RSs and Ubiquitous Computing (ubicomp) [3] have a common goal, which is to remain "transparent" and invisible to the users, making them interact with the system in a natural and immersive way. Ubicomp tries to study techniques aiming to integrate technologies into people's everyday life and has been considered the natural destination to which desktop computing tends to migrate, as described by [4]. Ubicomp is becoming increasingly widespread and accessible in several fields such as Logistics [5], Education [6], and Health [7]. However, Ubiquitous Accessibility (U-Accessibility) is an area that lacks studies and researches that bring emerging solutions, as highlighted by [4]. According to the reasoning approached by [4], following the behavior of classical computing, which now migrates from desktop to ubiquitous and cloud computing, it is necessary to break paradigms and look for solutions that not only "adapt the computer in front of us". This lack of research and solutions regarding accessibility brings together a vast and growing area.

The use of information from a user's history is already widely disseminated in some approaches of RSs, as in the collaborative approach. Thus, it is possible to analyze such information and use it in order to estimate a recommendation of items. This analysis may be performed by a similarity analysis, which measures a similarity index between two user histories. Information on a user's context history, also known as trails [8], differs from an ordinary history because it is grouped and organized in a chronological way, so that the whole set is analyzed and not only each history's element.

Based on this context, this work presents improvements to be performed in a recommender system for accessibility, named Vulcanus [9], which, through a trail similarity analysis, recommends accessibility

resources to the user, considering information of the user's context and profile. Vulcanus considers trail similarity analysis [8] [10] [11], context information [12] [13], and users' profiles [14] in the recommendation of resources that offer accessibility to the user. These resources are estimated as the most likely to be of the user's interest, according to the similarity between the resource trail that the user is generating and the trails already performed by the community having the same disability. Moreover, Vulcanus can be integrated into several projects aiming to promote accessibility to people, as in the Hefestos project [15]. The system is able to recommend resources that offer accessibility services to any type of disability, both physical and mental. As a continuation of previous work [9], this study presents an asymptotic analysis of Vulcanus' algorithm and proposes improvements to be performed, creating the Vulcanus 2.0.

Despite the fact that Vulcanus is able to encompass a variety of disabilities, we developed its prototype exclusively for motor disability, since the mapping of resources with accessibility to locomotion has an open community that maintains itself updated [16]. The use of real resources aims to spread accessibility to the users, offering resources used by other users with the same disability and in similar situations. In order to simulate trails performed by the community, which are used for trails similarity analysis, we implemented a trail simulator that, based on different user profiles, makes use of resources based on a probabilistic distribution. The simulation generated data trails for 300 wheelchair users, with three different profiles, during a period of six months. For the trail similarity analysis, we used two approaches, one considering trails of used resources and another evaluating trails of categories of used resources.

This work is divided in 6 sections. The section 2 presents related works and compare them according to their features and limitations. Next section 3 brings the Vulcanus model, its architecture and features. The section 4 presents the prototype implementation, the description of evaluation scenarios, and also the obtained and inferred results. The following section 5 describes the changes performed on Vulcanus' algorithm regarding performance. Finally, the section 6 concludes the study with its contributions and presents future works to be performed.

2 Related Works

As mentioned earlier, the use of RSs in the field of accessibility still lacks studies, but we selected works having some relation to Vulcanus. In fact, we did not find any work proposing to solve the lack of RSs designed specifically for accessibility, despite the existence of several studies that might use a RS in order to improve the promotion of accessibility.

We selected some related works because they developed some type of RS in the study and because they are directly related to disabled or elderly people and thus affect people with special needs. The works were evaluated according to the classification of the used RS, the target public of users, and the use of user profiles, contexts, history or trails, and similarity analysis.

2.1 Recommendation of Travel Plans for Elderly People

The study presented in [17] proposes an RS for travels that is exclusive for elderly people who usually have capacity and health restrictions. The work brings this solution, since current travel services do not consider such restrictions. The work aims to use Semantic Web to perform a complete survey on the restrictions and preferences of a group of users. The work presents a method of collaboration between requisition, profile and tourism models to elaborate a recommendation, and allows selecting a travel plan according to the group capacity.

Focused on the RS, it aims to filter, in an intelligent way, a set of destinations in order to attend to the group's preferences. These preferences may be obtained on each user's profile or be deducted from the user's consumptions. The study distinguishes four recommendation approaches:

- Stereotype approach: It classifies the user profiles in a list of categories. Each category has its own destinations. This approach is not used when the categories of user profile or destination classes are not defined and enumerated.

- Contains approach: This approach is based on the destination's content. The destinations already visited by the user are verified and so similar destinations are proposed. This approach is impracticable when a new user profile is created, as it does not have previous history information.

- Collaborative approach: In this approach, the user history and his or her evaluation are used to perform the calculation of the nearest user, thus generating a recommendation based on the history of the community of users. As in the Contains approach, when there is no history, this approach cannot be used.

- Knowledge approach: It consists in determining the combination between destinations and users. Knowledge may be defined (A) mathematically, using learning machines, neural networks, among others; or (B) implicitly, by using reasoning engines, such as the Semantic Web technology.

The RS of this work uses the travel history of each user, thus recommending other destinations having features similar to the ones that were already visited. In this case, the services are the offered items, while the user's capacity is a parameter of the filter. The user profiles, together with the tourism ontologies, give the RS the necessary information to elaborate the recommendation. The application has not yet been evaluated, but this evaluation will be performed in future works.

2.2 SOLVE-D

SOLVE-D [18] is a context-sensitive recommendation framework of personalized services for disabled people in smart home environments. As in the study of [17], SOLVE-D uses ontology for representing its data, dividing them into three main modules:

- Generic Service Ontology (GS-ONT): It is responsible for the ontology of standard domestic services. It is built using the knowledge in the manuals of each service provided by the manufacturers and interviews with the developers. This ontology has several concepts related to the standard processing of these domestic services. These services include, for example, washing machine, refrigerator, air conditioner, among others.

- Personalized Service Ontology (PS-ONT): This ontology is generated for the recommendation of personalized services, since GS-ONT has difficulties in recommending services based on the user's behavior or preference. If the user modifies the standard process of a service, this action is considered a signal that reflects his or her preference, conducting the task to a specific context. Therefore, this ontology undergoes changes according to the changes performed by the user, altering the recommendation of service processes.

- Service Context Ontology (SC-ONT): It represents context information. This information is obtained when the user changes the service's standard process, and personalizes it. These data are collected using sensors in the service (weight of dirty laundry in the washing machine, temperature in the environment of the air conditioner, etc.), as well as using Web Services (time, climate, temperature, among others).

The module that generates the recommendations is named PSRM (Personalized Service Recommendation Module). The module makes recommendations considering the user's profile and his or her context. With such information, the service's operation process considered more adequate to the context will be the recommended model. Figure 1 shows the search algorithm of the customized process in the ontology PS-ONT where each already stored context is compared to the current one and the most similar is then recommended.

The study was applied in a practical way, the developed framework was implanted in household appliances and a case study was performed. The study simulated a blind person trying to use the washing machine. Changing the service's standard process, new parameters were established for the activity and the framework was able to adapt to the new modification, adding it as a new possible process to be recommended based on context information obtained in the situation.

2.3 Project SAID

The project SAID (Social Aid Interactive Development) [19] proposes a tool of personalized interfaces that allows internet access to elderly people. The project was proposed based on the assumption that the structure of webpages is complex and that elderly people have difficulties to access the information. Thus, SAID aims to reduce the complexity of webpages. One of the study's premise is to adapt webpages according to the user's preferences, thus making it easy for the user to achieve his or her goals when accessing a given information.

This work uses recommendation algorithms based on probability that estimate the user's objective, offering the option to consult it quickly. This algorithm uses previously collected information and adapts according to the behavior of an observed user. The website information is organized according to the ontology presented during the study. The main source of information on the interests of each user is obtained through questionnaires filled previously to the use of the system SAID. Besides the information collected by questionnaires, the system also collects the number of times a user selected a specific context, showing interest on that information, and the time the user spends on the webpage, another indication that user has interest on that specific content.

Procedure Finding optimal PS-ONT

Begin process

```
i, j, k = 1;
for each PS-ONT_i
  for each context data con_(i, j)
    for each new context data con_(new, k)
      if (con_(i, j) .EQ. con_(new, k)) then
        SimVal_i = SimVal_i + 1;
      else k++;
    end for;
    j++;
  end for;
  i++;
end for;

PS-ONT_opt = max (SimVal_i);
```

End process

Figure 1: Algorithm for search of optimal PS-ONT. [18]

The study compares the use of the recommendation algorithm to Bayesian networks. However, in the presented situation, the use of Bayesian networks is invalid since it handles sub-trees independently in relation to the whole tree. Therefore, an approach using a RS provides an optimized response to the prediction of the user's interests and to recommend items according to a given subject.

2.4 User Modelling Wizard for People with Motor Disabilities

The work presented in [20] reports the development of user modelling wizard that is specific for people with motor disabilities. The wizard is used in order to obtain a deep understanding of the interaction patterns between users with motor disabilities and mobile devices, leading to a user model. Using this model, it is possible to determine configuration parameters of an application, customizing it specifically for the user.

Table 1: User Model Parameters. [20]

No.	Parameter	Description
P1	Preferred input device	Input by touch or switch
P2	Reachable areas	Screen areas that the user can reach
P3	Minimal size of elements in GUI	Minimal size of GUI elements so that the user can touch them
P4	Detects touch-down or touch-up	Interface detects when the finger touches the screen or is suspended
P5	Swype and Pan ability	Measures the ability to use swype and pan gestures on the mobile device
P6	Number of switches	Defines the number of switches that the user operates
P7	Hold-Time	Minimal duration of the input signal
P8	Lock-Time	Time period in which no input is performed
P9	Scan-Time	Time period after which the focus moves to the next element
P10	Touching ability	Evaluates whether the user is able to use touch as an input method

Using a context-based approach, the wizard uses recommendation algorithms to determine configuration parameters that define the interaction of the user with mobile devices. The determined parameters for the composition of the user model are described in Table 1. All parameters are related to the user's interaction with the device, since the interaction mode with the GUI (Graphical User Interface) varies according to the motor disabilities of each user.

According to the tests performed in the study, it has been proved that the configuration parameters proposed by the RS can compete with the parameters generated by the consultant. The next steps of the study suggest the development of an adaptation model that is able to change the visual representation according to the user's abilities. Moreover, a user model in execution time is proposed, being able to detect deviations and update them in real time.

2.5 Comparative analysis of the related works

Table 2 compares the works regarding the classification of the used RS, support system, and the use of the user profiles, contexts, history or trails and similarity analysis.

Regarding the RS's classification, the studies concentrate on different domains, then different types of RSs are already expected. We highlight the work [17], which uses a hybrid approach, having more resources to generate recommendations. While other works use a specific RS approach to provide the recommendation.

Table 2: Comparison between the selected works

Attribute	Travel Plans for Elderly People	SOLVE-D	Project SAID	User Modeling for People with Disabilities
Classification of the used RS	Hybrid (Collaborative, Content-based and Knowledge-based)	Context-based	Probability-based	Context-based
Support to users	Elderly	Disable and elderly	Elderly	Disabled and elderly
Context use	No	Yes	Yes	Yes
User profile use	Yes	Yes	Yes	Yes
History or trail use	Only history	Only history	None	None
Similarity Analysis use	Yes	Yes	No	No

Regarding the target users of the systems, the works [18] and [20] have wider approaches, supporting people with disabilities, as well as elderly users, while [17] and [19] focus their efforts specifically on elderly people. Among all works, only [17] does not make use of context information. However, as the others, it makes use of the user profiles for the personalized recommendation of items.

Regarding the use of trails, none of the works makes use of them. The works [17] and [18] make use of information from the user history, but this information is only analyzed individually and not in a collective or grouped way, which is the basic characteristic of trails. In the last evaluated issue, the same studies [17] and [18] use similarity analysis in their RSs, comparing the user's information with a different stored information.

From this analysis, we recognized the need of an RS for accessibility that considers aspects of contexts, user profiles, trails, and similarity analysis in order to provide to these people with special needs the resources that provide accessibility and convenience in their daily live. The proposed RS uses a hybrid approach that uses information from the community together with information of the items themselves.

3 Recommendation Model Vulcanus

In [9], we proposed Vulcanus, a recommender system for accessibility based on the trail similarity analysis. The system considers the resource trail that the user covered in order to make the recommendation of items that other users of the community have used. Therefore, the system promotes accessibility, as it highlights resources that other users used in a similar context, recommending them. We consider the community's opinion as unanimity, i.e., the resources and interests of the community are known as absolute truths. So the recommendation is elaborated based on trails for community, where we analyze the user's trail with all community's trails. Thus it is possible to infer that the more a resource is used, the more useful it is and the better they offer help to the user's accessibility. The next subsections present the main concepts involved in the study and the proposal of Vulcanus.

3.1 Concepts involved

Among the main concepts used in this work there are: User Profile, which represents the people that use the accessibility resources; Context, where the information is gathered from the environment in which the user is, e.g., his or her physical location (GPS coordinates); Trail, which is the chronological history of items used by the user; Resources, which are the items that offer some type of accessibility to the user, described in two categories: (1) Generic, which attends to all types of disabilities, or (2) Specific, which only attends some disabilities; Disabilities, which are the needs that each user has. The following items report the used concepts and its features:

- User profile: It represents the entities that need the resources with accessibility. It may be a person with a disability or an elderly person with different special needs. The profile contains information about the user, such as name, age, disability and login data. Vulcanus uses this information in order to perform a recommendation. The information on disability is indispensable, as the recommended resources are directly correlated to the type of disability of the user. For example, if the user has a visual impairment, items that do not give accessibility to this type of disability should not be recommended, such as an access ramp. The user has an account that contains all this information, which is maintained in a database.

- Context: This represents the context information in which the user is found. Vulcanus performs the recommendation of resources to the user using this information as a reference. Such information is related to location, used resources and the user's disability. Among all the trails generated by the community, the similarity analysis finds the most relevant trails according to the trail generated by the user, thus recommending to the user the next resources that follow in the trails of the community.

- Trail: It represents the resources that each user consumes throughout the day, thus maintaining a chronological history of the resources' usage, storing together all the context information in which the user was while using the resources. With this chronological information of the trail, the RS can analyze the similarity with other trails of the community that happened in a similar context. From that, it is possible to perform the recommendation of items that are ahead in the community trails, inferring information based partial user trail.

- Resources: This corresponds to the items that the RS recommends to the user. The resources may be generic or specific, i.e., they give accessibility to all disabilities or are specific and give support only to some types of disability. For example, a parking space is a generic accessible resource, as it attends to all disabilities. On the other hand, an access ramp is a specific resource, as it only attends users with motor disabilities, so it does not give support to others.

- Disabilities: It refers to the special needs that each user has. They have multiple classifications, such as physical, visual, auditory, mental, among others. The disabilities become a type of filter in the trail mining. This filter occurs because, for example, only the trails of the community of wheelchair users are relevant for a wheelchair user. There is no evident benefit in analyzing trails of different disabilities. Thus, it guarantees that only the resources that in fact offer support to a given user's need are recommended.

3.2 Architecture of the model

Vulcanus is able to generate recommendations of items that have accessibility according to two approaches of trails' similarity. One considers the use of the resources themselves and the other analyzes based on the resources' categories in the trail. While the former evaluates the chronological sequence of resources, the latter analyzes the sequence of categories of resources, even if the community has not used any resource in specific from the user's trail. Figure 2 shows the modelling of Vulcanus and its modules. Vulcanus uses user profiles [14], context sensitivity [13], trails [8], mapped resources [16] and similarity analysis [10].

3.3 Components

3.3.1 Profile module

In this module, it is proposed that each user stores his or her profile not only with identification data, but also with information that refers to the type of disability and special needs. This module is used in the resource recommendation (together with the other modules) since the profile information is considered critical for the resource recommendation. Since each recommendation is intended to a user, different needs must be considered. This user profile information is used in the contextualization of the recommendation. As

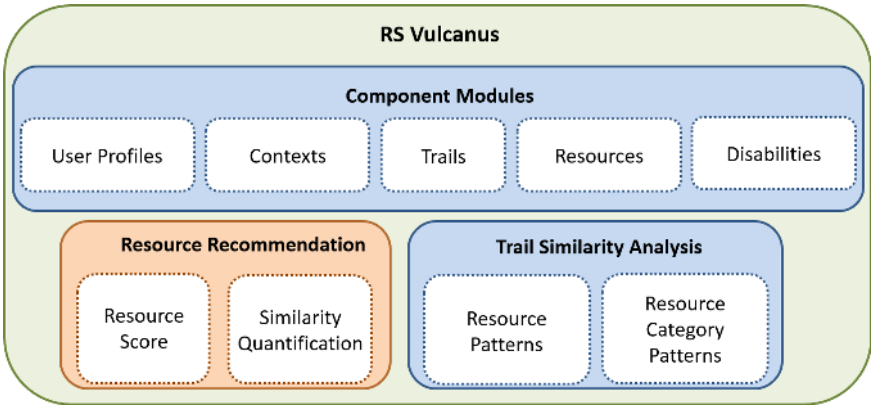

Figure 2: Vulcanus model.

presented in Figure 3, the profile information is used in both *Disability Context*, which correlates a disability to the user, and the *Temporal Context*, which registers the events performed by each individual.

3.3.2 Context module

This module is responsible for the storage and maintenance of contexts, as shown in Figure 3. Three different types of context information are used: *Location Context*, *Disability Context* and *Temporal Context*. The *Location Context* encompasses the information of user's location (GPS coordinates, or latitude and longitude), place name, ID (Identification), among other information of the environment in which the user is might be located. The *Disability Context* represents the disabilities that the user has and the disabilities in which each resource is able to provide support. Finally, the *Temporal Context* is responsible for the events that occur in the contexts, such as the use of resources or the negative indication to a resource available in the environment. An example of negative evaluation would be the indication that a mapped resource actually does not exist or is not correctly situated. In this last type of context, the type of event, name, description, hour, date and an ID are stored.

3.3.3 Trail module

This module manages all the trails that a user has. The trails can be defined as a series of "snapshots" containing information from context, profile, event, and used resource obtained over time according to the user's actions. All information necessary for the creation of a trail is show on Figure 4. The profile information reveals the needs that must be assisted; the resources provide accessibility for disabilities and the location of the event's occurrence; and the event refers to the classification of the situation. All this information, together with the *Context Module*, generates each element of the trail, which is composed by a collection of those. Furthermore, the similarity analysis is performed between the trail generated by the user and the community's trails. Each trail event has the following information: date, time, user ID and resource ID (*Disability Context*). The information of the *Location Context* may be inferred by the resource location itself, approaching also situations where the resource could move. For instance, in the case of accessible buses with multiple stops in their routes.

3.3.4 Resource module

This module is responsible by the management of resources, their mapping, classification, and to which needs they are destined. The resource may be either mobile or static, generic or specific. All this information, as well as name, description, location and an ID, is stored and managed in this module. A resource might be considered as any item that provides some type of accessibility for at least one disability. Some examples or resources are establishments with access to wheelchairs or other disabilities, access ramps, parking places, elevators, sound signals, and notifications in Braille, among others. The resources may be classified in several categories. In the example of Figure 5, the resource was classified in the category *Food* with the subcategory *Cafe*. Besides this information, its location in coordinates and which disabilities the resource support are also maintained. In the presented case, all resources present support to wheelchairs, which is the disability approached in the evaluation model.

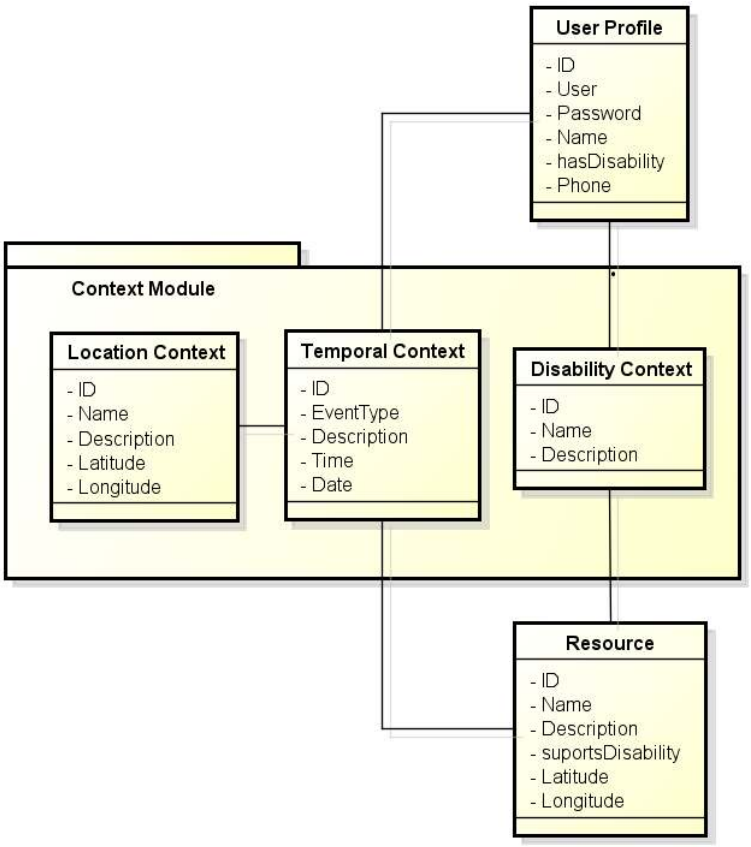

Figure 3: Diagram of the context module.

3.3.5 Similarity module

This module evaluates the similarity of the user's generated trail or it is still generating with those generated by the community with same disability, so being able to find similar trails. Based on this analysis, a trail list is generated. This trail list contains all relevant trails for the recommendation of items and will be subsequently used in the *Recommendation Module*. Figure 6 presents a sequence of actions performed for the mining of relevant trails. This module acts as a filter, where from all the trails generated by the community, only certain relevant and similar ones pass to the *Recommendation Module*. Trails are considered relevant if generated by users with the same disability as the user generating the trail. The similarity analysis only occurs on relevant trails, so there is a filter to reduce the amount of trails to be compared. The concept of similar is wider, as it considers information of the trail that the user is generating as well as information of the user's location.

3.3.6 Recommendation module

In this module, the relevant trails filtered by the *Similarity Module* are evaluated. The resources of these similar trails receive a proper score. This score is calculated through the amount of elements that are equal to, or belong to the same category (depending on the approach used), the trail generated by the user in relation to the community trails. Thus, the resources with higher scores that are ahead in the relevant community trails are recommended.

Figure 7 presents the sequence diagram of the recommendation. The implemented recommendation uses two approaches: one that analyzes the resources' trail and other that explores the trail of resources' categories, i.e., not the resource itself, but rather the category of each resource in the trail. This second approach also considers situations where the user is generating a trail with resources that have not been yet used by the community. In this specific case, an analysis of the resources' trail is not applicable. Instead, this approach explores the resource categories, identifying all trails that have resources of a specific category and there are in a specific range from the user's location. Therefore, even if the user is using a resource so far unknown from the community, a recommendation will be offered approaching the trail of resources' categories, which is always able to perform a recommendation of resources to the user.

The implemented recommendation needed three loop levels (Figure 7): one for the size of the trail

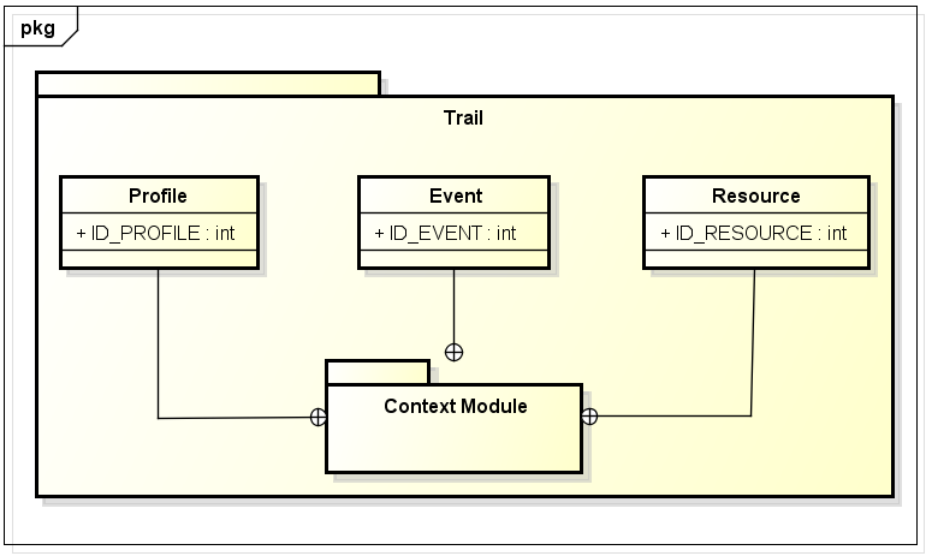

Figure 4: Internal components of a trail.

generated by the user, other for its comparison to each relevant trail and the last one for performing the comparison itself. After this evaluation, the subsequent resources of the community's trails are recommended and ordered according to their score. The higher the similarity, the higher the score; the higher the frequency that a subsequent resource appears, the higher its score.

4 Aspects of Implementation and Evaluation

4.1 Implementation

In order to evaluate Vulcanus, we implemented a prototype that consists of the several modules of the system. All the used data and information were stored in a database exclusive for the application. The chosen database was MySQL Workbench 6.1, a simple database of easy implementation, but appropriate enough to be used in the proposed prototype.

Information of 300 fictional wheelchair users were stored in the database developed in the prototype, as well as 8756 establishments with accessibility to this community of disabled people and more than 800 thousand resource events used in the trails, generated from a simulation that inferred the use of resources by the 300 users during a period of six months. The database followed the diagram in Figure 8, which contains all the components presented in the model's description.

The table *Disability* contains all the disabilities that the users of all communities have. The table contains the ID, name and description of each of the disabilities. Thus, it is possible to optimize the size of data, since the users with the same disability will have the information based on the ID and not all the replicated information. On the developed prototype, only one disability, wheelchair, was since we aimed a simple application for the evaluation of Vulcanus model.

The table *Profile* stores the information of each of the system's user. Information such as ID, name, login, password, telephone and disability (only its ID) are essential for obtaining information of context, insertion of events in trails and for the recommendation itself. We created 300 fictional users, all with the same motor disability, as they were all wheelchair users.

The table *Resource* presents all the resources registered in Vulcanus. Each resource has an ID, name, type, category, latitude, longitude, and the disability to which it gives accessibility (only its ID). We recorded 8756 real resources, obtained obtained through the open community Wheelmap [16]. In this open community, each user may record real establishments in his or her city and evaluate them according to the accessibility provided to wheelchair users. There are four levels of category: fully accessible, partially accessible, non-accessible and unknown. All 8756 recorded establishments are fully accessible, as this is the only classification that is relevant for Vulcanus. According to Figure 9, these resources are located in the city of Berlin, Germany.

These resources are divided in 12 main categories. These categories are attributed during the creating of each establishment. These categories are information of high value, because it is through them that the approach of category trails is implemented. Furthermore, this information have also become relevant in the creation of the community trails due to the approach of different profiles with different needs. The categories

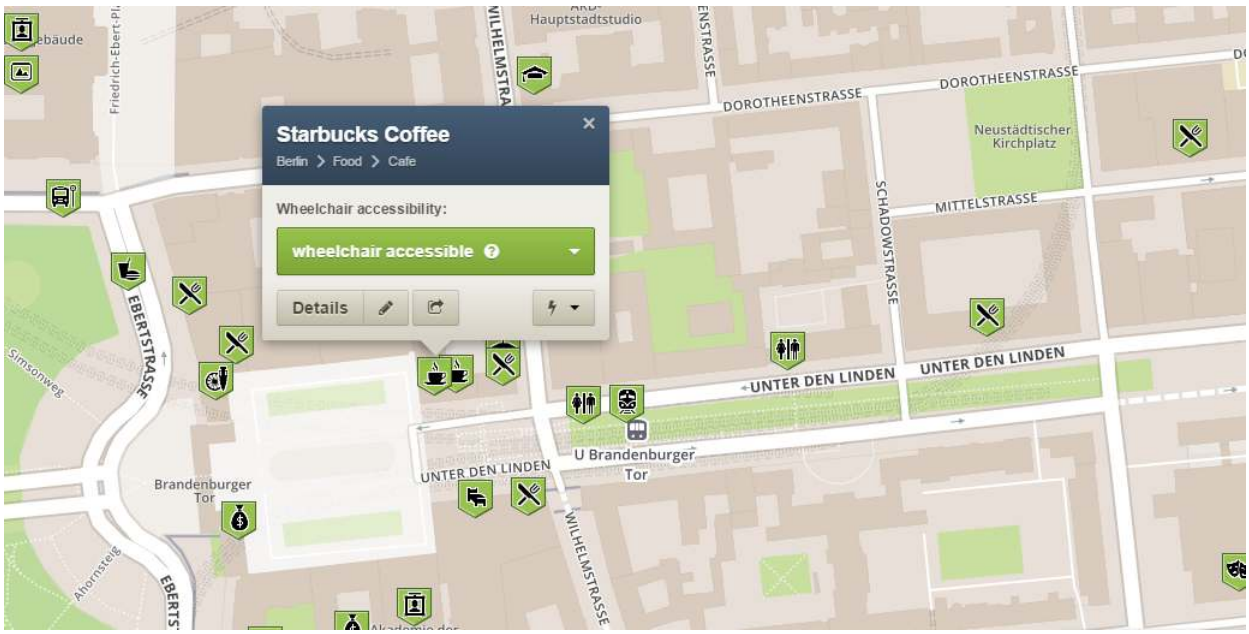

Figure 5: Resources and their information.

used in the resource classification are listed in Table 3.

Table 3: Categories for resource classification.

Categories
1 - Public Transfer
2 - Food
3 - Leisure
4 - Bank Post
5 - Education
6 - Shopping
7 - Sport
8 - Tourism
9 - Accommodation
10 - Miscellaneous
11 - Government
12 - Health

This list of resources was obtained from an API (Application Programming Interface) of the community Wheelmap itself. In this API, a pair coordinates (*Point A* and *Point B*) form an imaginary rectangle where all the resources within this coordinates are returned in an XML (eXtensible Markup Language) archive. The used imaginary square covers the city of Berlin, beginning at the coordinates of latitude 52.569107 and longitude 13.215619 (*Point A*) to the final coordinate of latitude 52.381737 and longitude 13.716870 (*Point B*) presented by Figure 10. We chose the city of Berlin for the use of the prototype because it is the capital of Germany, which is the host nation of Wheelmap, having a huge number of registered resources. In addition, in the year 2013, Berlin received the "Access City" award, organized by the European Disability Forum [21].

Besides the resources, the nearest resources information was also stored. In the table *Resource_Near* the relationship of the 1000 nearest resources and each registered resource, with their respective distances, are stored. This information was necessary especially in the simulation of the community trails and was used in the similarity analysis that approaches the trails of resource categories.

Finally, the table *Trail* contains all the events of resource use. During the simulation, each time a user used a resource this event was stored. Each trail contains an ID, the user's ID, the used resource's ID, date, time, distance since the last resource used by the user until that moment, and the type of user in that given day.

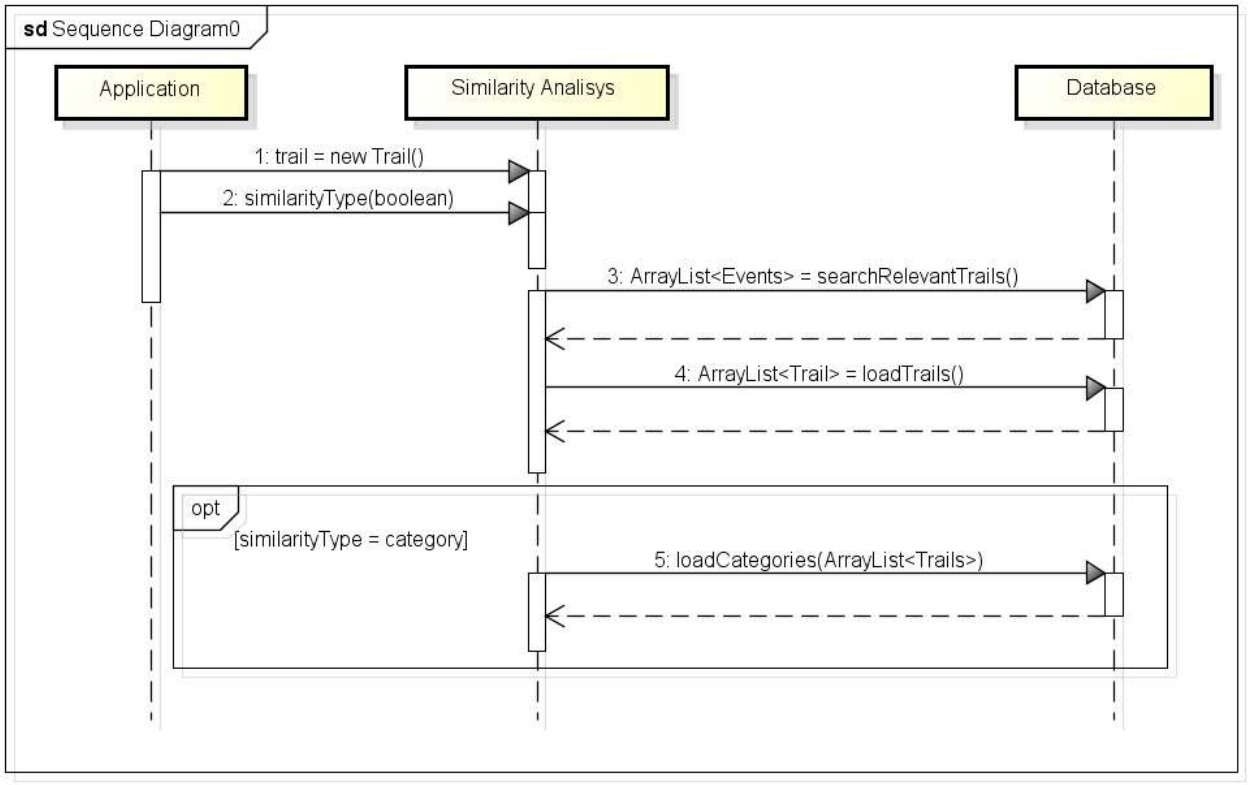

Figure 6: Diagram of similarity analysis sequence.

4.2 Trail simulation

Since the community trails, generated by the use of registered resources, are a kind of data that need to be collected or simulated, we evaluated and elaborated the implementation of a trail simulator able to generate logical and viable trails. These trail data cannot be generated randomly because, since they deal with real resources, the user locomotion is an important information to be considered. For example, if we consider a wheelchair user that used a resource in a part of the city and soon after used another resource 20 kilometers away and then returns to a resource close to the first one in a short time, this trail is clearly unreal and does not represent the user's behavior.

Aiming to create trails for the use in resource recommendation, we elaborated a trail simulator able to elaborate feasible trails simulating a real behavior. Using all data cited in the implementation of the prototype of Vulcanus, the simulator generated data from the 300 users for 180 days. In each day, each user adopted one of the three created profiles: *Worker*, *Tourist* or *Lazy*. In each of these profiles, the user is subjected to the use of a number of resources. On each day, the profile that a user will have is randomly chosen based on the probabilistic distribution presented in Table 4. In addition to the probability of each profile, we also defined the time windows in which the resources may be used. This information of "total hours" and "velocity" refers to the range of resources that the user reaches, which will be explained further.

Table 4: Probabilistic Distribution of Profiles in Everyday Use.

Profile	**Daily Probability**	**Daily Period**	**Total hours**	**Velocity**
Worker	65%	6 to 18h	12 hours	2 Km/h
Tourist	10%	8 to 22h	14 hours	4 Km/h
Lazy	25%	9 to 20h	11 hours	1 Km/h
Total	100%			

The number of resources that the user uses on each day is randomly generated, varying between 5 and 35 resources. Each profile has a different distribution that refers to each need. This distribution of the number of used resources on each day is presented in Table 5.

For each day, the number of resources to be used is defined. From this information, together with the total hours of the daily window of each profile, it is possible to know the interval of the use of each resource. Defining this interval as a value Δ, it is possible, together with the information of velocity (ν), to estimate

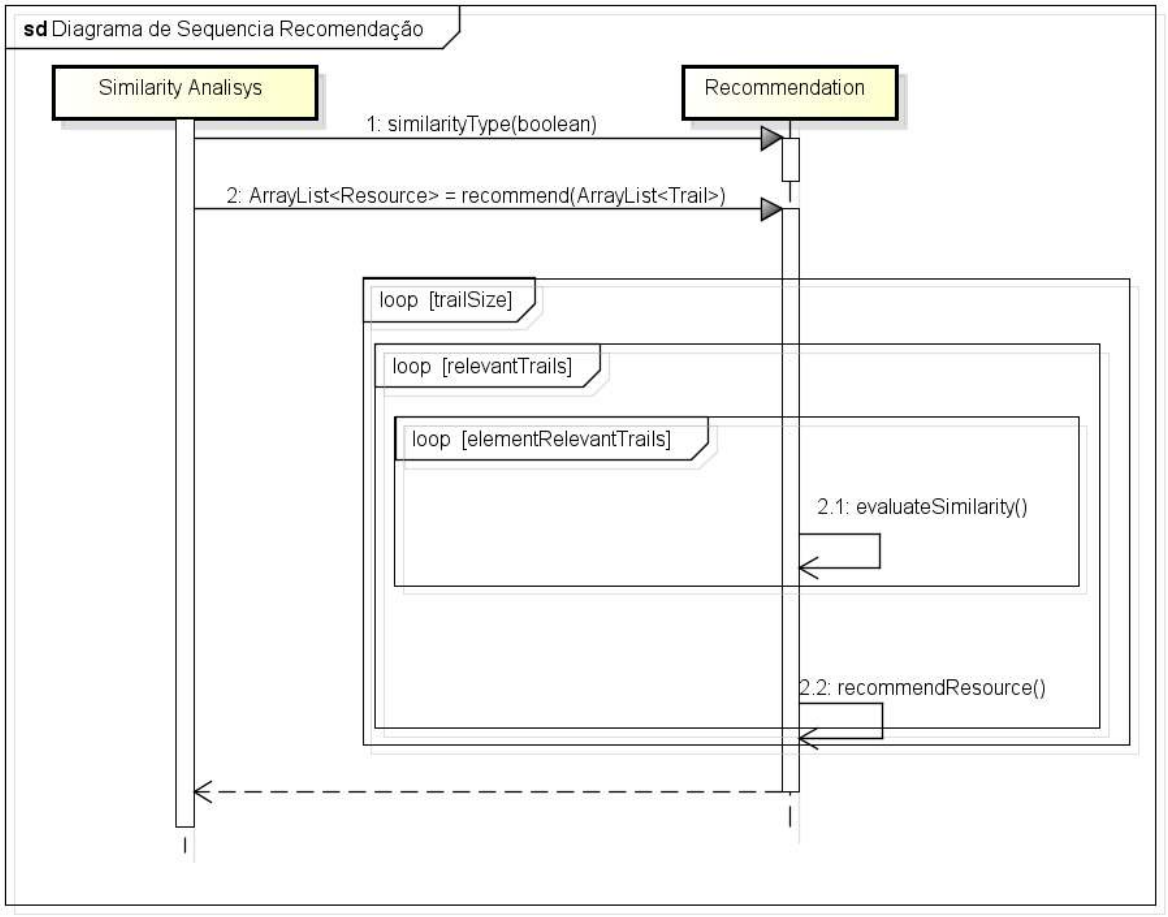

Figure 7: Diagram of recommendation sequence.

Table 5: Probabilistic Distribution of the Daily Number of Used Resources.

No. of Resources	**5∼10**	**10∼15**	**15∼20**	**20∼25**	**25∼30**	**30∼35**
Worker	5%	40%	40%	10%	2,5%	2,5%
Tourist	5%	5%	15%	30%	30%	15%
Lazy	80%	15%	5%	0%	0%	0%

the range (γ) for the use of resources on that specific day. This value is defined by the formula: $\gamma = \Delta \times \nu$. Based on this range value, the uses of resources are performed. Each user is randomly connected to one of the 8756 resources. This resource will be the user's starting point, which will daily begin with the use of this starting resource.

Moreover, during the trail creation, we considered the categories of each resource and attributed a probability of use of a certain category, given the user's profile in a certain day. This probabilistic distribution is shown in Table 6. It was defined by discussing among experts involved in the work. Some categories have a higher incidence in certain profiles, for example *Transport*, *Education*, *Health*, among others, which have a high incidence in the profile *Worker*. On the other hand, the profile *Tourist* has a higher incidence in the categories *Tourism*, *Leisure*, *Accommodation*, and *Shopping*. Finally, the profile *Lazy* has a higher incidence of the categories *Leisure*, *Food* and *Misc*.

4.3 Methodology

The research community has been using the scenario methodology for the validation of context-sensitive systems [22], ubiquitous systems [23] and systems of ubiquitous accessibility [15]. Similarly, for the evaluation of the Vulcanus model, we performed simulation in three different scenarios, each one describing a possible application of the system. So we provided an input (according to the scenario), and then we discuss the recommendation output of Vulcanus. Each scenario approaches three wheelchair users, in the city of Berlin,

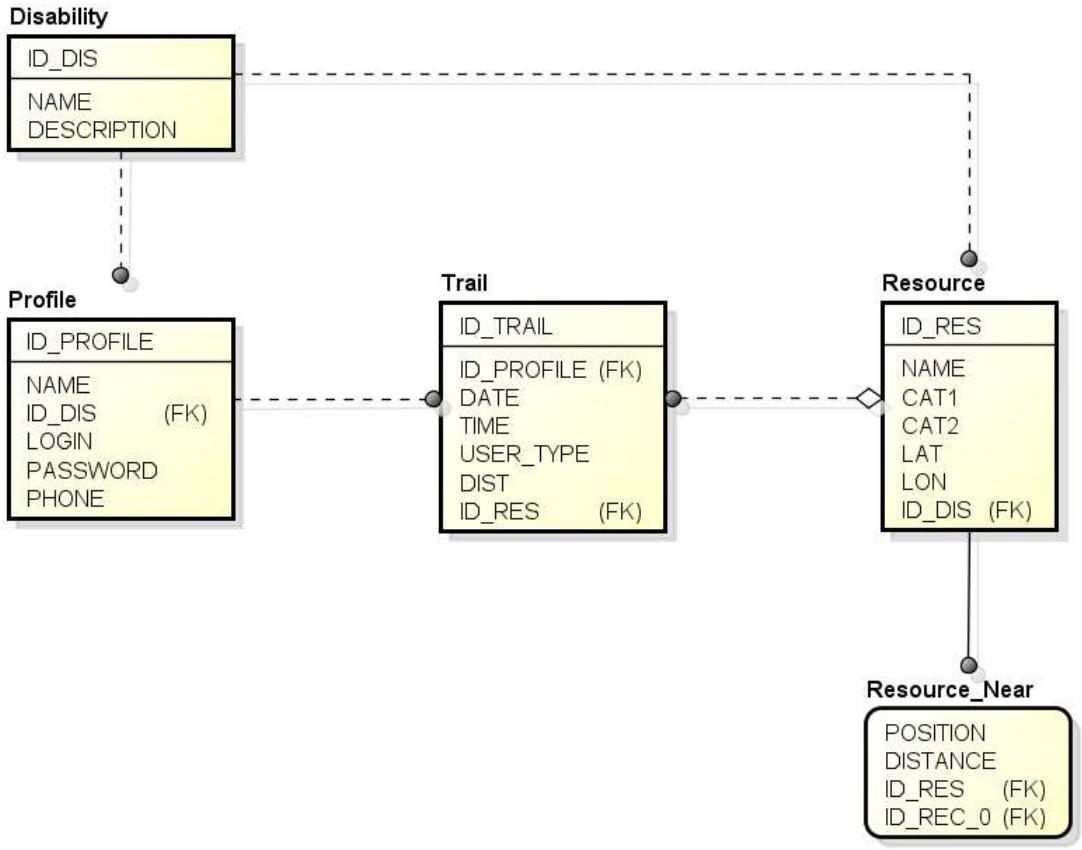

Figure 8: RE (Relationship Entity) diagram of the database used in the prototype.

in different contexts and with distinct purposes. The description of each scenario is given below:

4.3.1 Scenario 1: Tourism

"Amanda is a Brazilian wheelchair user and decided to spend her summer holidays in Germany with a companion. After making all reservations in Brazil, she and her companion travel to Berlin and disembarked at the local airport. After that, they take a wheelchair-adapted taxi, arrive at the booked hotel (Bax Pax Downtown Hostel/Hotel) and check in. At the hotel, the purpose is to have a good night's sleep before the next day's journey. The couple plans to visit several tourist sites. As they do not have any previous experience in the city, they use a tourist application for smartphones. This application was developed exclusively for tourists in Berlin and presents a recommendation of establishments with accessibility to wheelchairs. The recommendation system that integrates the application is Vulcanus RS, using an approach of resource trails. Leaving the hotel as the point of origin, the couple starts the day. Following the application's recommendation, the couple decides to visit the museum "Alte Nationalgalerie", moving around via taxi. After visiting the museum, they wish to have lunch in the Indian restaurant "Aari", and, for that purpose, they take the German public transfer, the "U Oranienburger Tor". After lunch, the couple search for a nearby ATM (Automated Teller Machine) in order to withdraw money and then, via taxi, go to the "Museum Blindenwerkstatt Otto Weidt"..." Table 7 presents a partial trail generated through a fraction of the couple's day.

4.3.2 Scenario 2: Worker

"Helen is a German wheelchair user searching for an easy routine. Despite living in Berlin for about two years, she still does not know many places that offer accessibility to her disability. Thus, after an indication of a friend who is also a wheelchair user, she installed in her smartphone an application of urban travelling that, more than only generating routes, also recommends establishments with accessibility to several disabilities. This application has the Vulcanus RS focused on trails of resources' categories and it is able to save the trails daily generated by the users. Helen intends to use the application in her daily life in order to know new available resources in her routine range, looking for convenience and accessibility to her special needs. Following her routine, she usually uses accessible public transfer to go to the school "Grundschule unter den

Figure 9: Map with resources used by the prototype.

Figure 10: Area with resource mapping.

Kastanien", her working place. At the school, during her lunch break, she usually goes to nearby restaurants and fast foods, returning to work after the meal. By the end of the workday, she returns home using again the public transfer. Based on this description of a daily routine, the trail in Table 8 was elaborated."

From this category trail, after a similarity analysis with other trails and the score of resource recommendation, the application used by Helen returned the establishments presented in Table 9.

4.3.3 Scenario 3: Lazy

"Thomas is a German wheelchair user looking for new places to have fun, both with friends and with his girlfriend. He is attentive to new technologies and read in a blog dedicated to assistive technologies about the release of a new GPS that includes recommendation of places according to the support to different disabilities. Interested in the product, he acquired the GPS and started to test it. This GPS includes Vulcanus RS, specifically its module of category trails.

During the weekend, Thomas and his girlfriend use the GPS and go to "Getränke Hoffmann" (Shopping - Drinks) in his private car. After arriving, the GPS generates a recommendation of several resources. The couple becomes interested in "Kino UNION Filmtheater" (Leisure - Cinema, Figure 11a)) and decides to go there. During the film, Thomas needs to go to the bathroom and finds its reference in the GPS, the recommendation of the resource "City Toilet" (Misc - Toilet, Figure 11b)). After using the toilet, the recommendations are updated (Figure 11c)) and he returns to the cinema. Back to the cinema, the recommendations are again updated (Figure 11d)) and the couple decides to return home."

Table 6: Probabilistic Distribution of Resource Categories in Relation to the Profile.

	Worker	**Tourist**	**Lazy**
Transport	**16%**	8%	3%
Food	**15%**	8%	**15%**
Leisure	1,5%	**15%**	**30%**
Money Post	**15%**	8%	3%
Education	**15%**	1%	3%
Shopping	1,5%	**10%**	3%
Sport	1,5%	8%	9%
Tourism	1,5%	**20%**	3%
Accommodation	1,5%	**10%**	3%
Misc	1,5%	8%	**15%**
Government	**15%**	1%	3%
Health	**15%**	3%	10%
Total	100%	100%	100%

Table 7: Partial Trail of Scenario 1.

Trail's Element	**Resource Name**	**Position in Recommendation**
1	Bax Pax Downtown Hostel/Hotel	X
2	Alte Nationalgalerie	3
3	U Oranienburger Tor	11
4	Aarti	7
5	ATM	5
6	Museum Blindenwerkstatt Otto Weidt	3

4.4 Results

Based on the scenarios described in Subsection 4.3, it is possible to perform an analysis and infer aspects in order to ascertain whether the study used the correct approaches, as well as to identify where the study must focus its improvements. The choice of different scenarios, with different focuses, aims to evaluate Vulcanus on different aspects. In each scenario, we analyzed and evaluated the prototype's behaviors in search for the system's negative and positive aspects.

The first scenario included the use of recommendations, which in turn were generated through the approach of resource trails. In scenario 1, the couple used resources listed in the recommendation, adding them to the trail generated in real time, an event that leads to the generation of a new recommendation. The couple had a tourist profile and the choice of resources was based on this specific interest. Leaving the hotel, the couple went via taxi to the museum *Alte Nationalgalerie*. Looking for a place to lunch, they took an accessible bus (*U Oranienburger Tor*). After that, they went to an Indian restaurant (*Aarti*) and then to an ATM. Finally, they visited the museum *Blindenwerkstatt Otto Weidt*, which finished the scenario's description.

The last column of Table 7 presents the resource's position in each of the generated recommendations. Through these indexes, it is possible to observe that some resources used by the couple, mainly the third and fourth resources (with positions 11 and 7, respectively), obtained a low degree of similarity with the trails generated by the community and therefore a lower score.

In the case of the third resource (*U Oranienburger Tor*), the system indicated ten resources of higher relevance based on the trail generated by the couple, even if these resources were not relevant to the tourism context presented in the scenario. In the second case, the fourth resource used by the couple (*Aarti*) was only in the seventh position of the recommendation generated by the system, which therefore had six more relevant resources, i.e., with higher scores.

After the evaluation, it was found that the community trails are determinant for the generated recommendations, because if the community trails do not make sense, neither do the recommendations. And even with the profile *Tourist* described in scenario 1, many resources of the profile *Worker* obtained a higher relevance in the recommendations due to the similarity algorithm, which was not developed to filter trails based on the profile of the user that generated it.

The second scenario approached a recommendation according to the category trail. After the presentation of the user's trail, a recommendation of resources was generated, i.e., the trail composed by *Public Transfer*, *Education*, *Food*, *Education* and *Public Transfer* was compared to all the community trails and, after a filter, it indicated the relevant ones, i.e., the trails having a resource of the category *Public Transfer* within

Table 8: Trail elaborated by Scenario 2.

Resource ID	Resource Name	Category	Type
6932	Kranoldplatz	Public Transfer	Bus Stop
6916	Grundschule unter den Kastanien	Education	School
6909	Burger King	Food	Fast-Food
6916	Grundschule unter den Kastanien	Education	School
6668	Lange Straße	Public Transfer	Bus Stop

Table 9: Recommended resources based on the category trail.

Recommendation Order	Resource Name	Category - Type
1	Deutsche Bank	Bank Post - Bank
2	Grundschule unter den Kastanien	Education - School
3	ATM	Bank Post - ATM
4	Milans	Food - Coffee Shop
5	Hedwig Apotheke	Health - Pharmacy
6	Berliner Sparkasse	Bank Post - ATM
7	Thai-Restaurant iMM-DEE	Food - Restaurant
8	PizzaLiebe	Food - Restaurant
9	Gertruden-Apotheke	Health - Pharmacy
10	Seydlitz Apotheke	Health - Pharmacy

a certain range from the user's location. These relevant trails are then evaluated, receiving a score according to the similarity level. For example, a trail containing *Food*, *Education* and *Public Transfer*, in this order exactly, has a higher similarity than a trail having *Education* and *Public Transfer* also in this order exactly when compared to the category trail of scenario 2. The generated recommendation is shown in Table 9, which informs the *Top 10* resources with higher relevance and similarity.

Given the scenario 2, the presented recommendation was in fact able to offer relevant resources that attend to the scenario's needs. From the 10 resources of Table 9, only the second one may be questioned, while the others were able to attend to the needs imposed by the scenario, which seeks resources near to the user's geographic location through daily services (for example, banks, restaurants and pharmacies). The second resource *"School - Grundschule unter den Kastanien"* may be questioned because it was already used during the generation of the category trail. This reveals that Vulcanus does not consider whether the resources were used along the trail generated by the user, thus being likely that it recommends a resource already known by the user, affecting the innovation in the resource offering.

Finally, the third scenario presented the recommendations generated at each new resource use. Each time the user uses a resource, the application detects this event and the resource is added to the trail that the user generates in real time. In scenario 3, such as in scenario 2, an approach of category trails was evaluated. The user started from a resource of the category *Shopping*, going to *Leisure*, which was the fourth resource in the list of recommended resources. The subsequent resources were from the categories *Misc* and *Leisure*, respectively.

Figure 11 presents the *Top 5* recommended resources. The user generates this recommendation list at each new use of resources, improving the generated trail. The recommended items are shown according to the user's described proposition, yet some resources, such as in scenario 1, are more appropriate for a profile *Worker* than a profile *Lazy*, which the user presents. That is the case with the resources *"Esso"* (*Public Transfer*); *"EKT - Friedrichshagener Kinderladen e.V."* (*Education*); and *"Bügeramt III"* (*Government*). This happens because the community trail is not filtered according to the user's profile. The user's profile (*Worker*, *Tourist*, and *Lazy*) was elaborated in the trail simulator, not in Vulcanus. Vulcanus considers the user's profile, but this information is specifically related to each individual's disability and needs.

Such as in scenario 2, which also uses category trails, scenario 3 presents repeated resources, which, despite having been used in the user's trail, are still recommended and presented in the *Top 5* list of recommended resources. Through this scenario, it was possible to analyze a peculiar behavior of the user, who repeats the use of resources in the same trail. This behavior ends up illustrating a different context, because in this

Resource ID	Resource Name	Type	Category
3748	City Toilette	Toilet	Misc
3765	Esso	Fuel	Public Transfer
1044	Berliner Sparkasse	Bank	Bank Post
6472	Kino UNION Filmtheater	Cinema	Leisure
3769	EKT - Friedrichshagener Kinderladen e.V.	Kindergarten	Education

6472	Kino UNION Filmtheater	Cinema	Leisure
3748	City Toilette	Toilet	Misc
1044	Berliner Sparkasse	Bank	Bank Post
6469	Kristinen-Apotheke	Pharmacy	Health
6563	Bürgeramt III	City Council	Government

6472	Kino UNION Filmtheater	Cinema	Leisure
3748	City Toilette	Toilet	Misc
1044	Berliner Sparkasse	Bank	Bank Post
3750	Bölsche Imbiss	Fast-Food	Food
6563	Bürgeramt III	City Council	Government

3748	City Toilette	Toilet	Misc
6472	Kino UNION Filmtheater	Cinema	Leisure
6469	Kristinen-Apotheke	Pharmacy	Health
3749	Zur Stammkneipe	Restaurant	Food
1044	Berliner Sparkasse	Bank	Bank Post

Figure 11: Top 5 recommended resources at each new trail element.

way, the recommendation of a previously used resource might becomes valid. The recommendation loses innovation, as the user already knows the resource. However, as this behavior is usual in the community, the RS correctly generates such recommendations.

5 Asymptotic Analysis

For Vulcanus, initially presented in [9], the main goal of its algorithm is comparing the sequence of element's categories between two distinct trails. For example, it is possible to compare a trail with following element's categories sequence: "Leisure(1) - Education(2) - Food (3)". Based on that sequence, the algorithm can identify trails with that same sequence. That sequence can be presented in any part of several different trails. After finding that sequence (completely or partially), the next resource of trail, if any, is added in a resources list and receives a score according to its incidence. The algorithm's results is a list of sorted resources, where the highest score refers to the resource which has a high incidence and/or similarity with the original compared trail. This is the first version of its algorithm:

```
1  // for different sizes of stack
2  for (Integer i = 0; i < stack.size(); i++) {
3      Integer c_stack = stack.size() - i;
4
5     // for every relevant trail
6      for (Integer j = 0; j < list.size(); j++) {
7
8
9          // for check every element of trail, that has stack length
10         boolean flag_same_cat = true;
11         boolean short_trail = false;
```

```
12
13          Integer index_trail;
14          Integer index_stack;
15
16          for (Integer k = 0; k < c_stack; k++) {
17
18              if ((list.get(j).getIndex()-k) >= 0){
19
20                  index_trail = list.get(j).getIndex() - k;
21                  index_stack = stack.size() - k -1;
22
23                  String comp_a = list.get(j).getArray().get(index_trail).getCat();
24
25                  String comp_b = stack.get(index_stack).getIden2();
26
27                  if (!comp_a.equals(comp_b)) {
28                      flag_same_cat = false;
29                      break;
30                 // BREAK when trail's element and stack's element are the same
31                 // the trail is relevant for score
32                  }
33              }
34              else{
35           // Check if trail is smaller than resource's stack
36           // Trail cannot be checked, too short
37                  short_trail = true;
38                  break;
39              }
40          }
41
42        //In case trail was checked, is relevant, then calculate score
43          if (flag_same_cat && !short_trail) {
44
45              Integer index_ = list.get(j).getIndex();
46              int siz = list.get(j).getArray().size();
47
48              if ((int) index_ < (siz - 1)) {
49                  Integer aux_res =
50               list.get(j).getArray().get(list.get(j).getIndex() + 1).getRec();
51                 //Check if resource was added into recommendation list
52                      if (!contemRes(res2Rec, aux_res)) {
53                            //First occurrence
54                       Recommend aux_rec = new Recommend(aux_res,score[c_stack - 1]);
55                             res2Rec.add(aux_rec);
56                             System.out.println(j.toString() + " - " + aux_rec.getRec().toString() + "
                                   got " + score[c_stack - 1].toString() + " It has " + c_stack + "
                                   element(s)");
57                         }
58                      else {
59                        //repeated occurrence, no need to add in list
60                          int addScore = score[c_stack - 1];
61                          adicionaScore(res2Rec, aux_res, addScore);
62                          System.out.println(j.toString() + " - " + aux_res.toString() + " got " +
                                score[c_stack - 1].toString() + " repeated " + c_stack + " element(s)");
63                      }
64              }
65          }
66          else {
67           //Do not calculate score
68              System.out.println(j.toString() + " - " + c_stack.toString() + " - No score");
69          }
70      }
71  }
```

5.1 Big O

We performed the asymptotic analysis [24] of first version of Vulcanus algorithm, by dividing the source code into small parts, that summed provide the total algorithm complexity. The following information is related to the complexity of each small parts.

- IF line 18-33
  18 (2 × Op. Get) + Op. Sub + Op. Comp
  20 Op. Attr + (2 × Op. Get) + Op. Sub
  21 Op. Attr + Op. Get + (2 × Op. Sub)
  23 Op. Attr + (4 × Op. Get)
  25 Op. Attr + (2 × Op. Get)
  27 Op. Equ
  28 Op. Attr
  ELSE 34-39 is not evaluated since its complexity is smaller than IF 18-33.

- FOR line 16-40
  16 Op. Attr + Op. Comp + Op. Add
  (IF 18-33)

- IF line 43-65
  43 2 × Op. Comp
  45 Op. Attr + (2 × Op. Get)
  46 Op. Attr + (3 × Op. Get)
  48 Op. Cast + Op. Sub + Op. Comp
  49 Op. Attr + (6 × Op. Get) + Op. Sub
  52 Op. LSearch
  ELSE 58-63
  IF 52-57 is not evaluated since its complexity is smaller than ELSE 58-63

- ELSE 58-63
  60 Op. Attr + Op. Sub
  61 Op. LSearch + Op. Add
  62 Op. Prnt + (3 × Op. ToStr) + (7 × Op. Soma) + Op. Sub

- FOR line 6-70
  6 Op. Attr + Op Comp + Op. Soma
  10 Op. Attr
  11 Op. Attr
  FOR 16-40
  IF 43-65
  ELSE 66-69 is not evaluated since its complexity is smaller than IF 43-66

- FOR line 2-71
  2 Op. Attr + Op. Comp + Op. Add
  3 Op. Attr + Op. Get + Op. Sub
  FOR 6-70

In summary, from all operation, we have a critical set which is defined by: FOR 2-71 × FOR 6-70 × FOR 16-40 × IF 18-33. Where the Big O(n^3). The described operations are the following: Attribution, Comparison, Addition, Subtraction, method get, method equals, method toString, method println, and Linear search in array. The terms that compose the Big O are related to the size of resources' stack, and the quantity of relevant trails for a determined sequence.

5.2 Dynamic Programming

Following the asymptotic analysis performed, we applied a dynamic programming technique (*bottom-up*) in order to improve the Vulcanus' algorithm. The new version of Vulcanus' algorithm is as follows:

```
// for different sizes of stack
for (Integer i = 1; i <= stack.size(); i++) {
    Integer c_stack = i;

   // for every relevant trail
    for (Integer j = 0; j < list.size(); j++) {

    if (!(list.get(j).getRelevance())){
            break;
      }

        // for check every element of trail, that has stack length
        boolean flag_same_cat = true;
        boolean short_trail = false;

        Integer index_trail;
        Integer index_stack;

        for (Integer k = 0; k < c_stack; k++) {

            if ((list.get(j).getIndex()-k) >= 0){

                index_trail = list.get(j).getIndex() - k;
                index_stack = stack.size() - k -1;

                String comp_a =
    list.get(j).getArray().get(index_trail).getCat();
                String comp_b = stack.get(index_stack).getIden2();

                if (!comp_a.equals(comp_b)) {
                    flag_same_cat = false;
               list.get(j).setRelevance(false);
                    break;
               // BREAK when trail's element and stack's element are the same
               // the trail is relevant for score
                }
            }
            else{
         // Check if trail is smaller than resource's stack
         // Trail cannot be checked, too short
                short_trail = true;
                break;
            }
        }

      //In case trail was checked, is relevant, then calculate score
        if (flag_same_cat && !short_trail) {

            Integer index_ = list.get(j).getIndex();
            int siz = list.get(j).getArray().size();

            if ((int) index_ < (siz - 1)) {
                Integer aux_res =
             list.get(j).getArray().get(list.get(j).getIndex() + 1).getRec();
               //In case trail was checked, is relevant, then calculate score
                    if (!contemRes(res2Rec, aux_res)) {
                          //First occurrence
                     Recommend aux_rec = new Recommend(aux_res,score[c_stack - 1]);
                        System.out.println(j.toString() + " - " + aux_rec.getRec().toString() + "
                            got " + score[c_stack - 1].toString() + " has " + c_stack + "
                            element(s)");
                               res2Rec.add(aux_rec);
                     }

```

```
63                  else {
64                 //repeated occurrence, no need to add in list
65                      int addScore = score[c_stack - 1];
66                      adicionaScore(res2Rec, aux_res, addScore);
67                      System.out.println(j.toString() + " - " + aux_res.toString() + " got " +
                            score[c_stack - 1].toString() + " repeated " + c_stack + " element(s)");
68                  }
69              }
70          } else {
71           //Do not calculate score
72              System.out.println(j.toString() + " - " + c_stack.toString() + " - No score!");
73          }
74      }
75  }
```

The first change performed in code is in the first FOR definition (FOR 2-75). Now that FOR has the initial value 1 instead of 0, which is going to be directly assigned for variable c_stack (line 3). That way, we changed the order of initial comparison. Before that change, the comparison was initially with the two complete trails. From now on, it initializes from the shortest, the element in trails' tail, and the following resources are added in every FOR iteration, until the final comparison with both complete trails' sequence.

In order to get a true search optimization, in line 32 we changed a boolean value, which stores information related to trails' relevance. This flag is modified to false in first occurrence when resources comparison is negative, which means that the first elements difference is found. From that point, the trail is no longer relevant, and its further comparisons with remaining elements of this same trail can be skipped. In lines 8-10, we included an IF condition which checks if the trail to be compared is relevant or can be skipped. All the trails are initially defined with relevance flag setted with value true.

6 Final Considerations

This study approached the implementation and optimization of Vulcanus[9] a recommender system dedicated to accessibility and focused on the recommendation of resources that give support to users with certain needs. We implemented an evaluation model that, through the trail similarity analysis, is able to recommend resources using two distinct approaches: resource trails and resource category trails.

6.1 Conclusions

The studies on recommender systems and their applications are already widespread. However, the area of accessibility, whether ubiquitous or not, still lacks researches. In other words, there are works proposing the development of assistive technologies for users with special needs, but such studies use recommender systems in a simplified form. This work was concerned about searching for RSs, their categories and applications in order to have a widespread theoretical base consistent with other researches that also developed and implemented RSs. Thus, this study presented the Vulcanus prototype as well as performance improvements in its processing time, offering relevant resources for users with special needs, creating the improved Vulcanus 2.0.

The results obtained in the developed scenarios confirmed that the recommendation offers resources that are in fact relevant to the users in the presented contexts, both in the resource trails and the category trails approaches. Scenario 1 presented recommendations according to the resource trail. Although the recommendation presents resources that are interesting to the user, the recommended resources did not achieve the highest scores due to the community trails, since the elaborated simulation contains trails that does not imitate a real behavior. Scenario 2 exhibited the recommendations according to the given trail, approaching resource category trails. This recommendation presents resources that were relevant to the user, even though there was among them a resource already used by the user in her trail. What seemed to be a failure, scenario 3 proved to be the opposite. Scenario 3 showed recommendation based on category trail according to the insertion of new resources in the user's trail. Contrary to scenario 2, in this case the user repeated the use of a resource, making the recommendation of a previously known resource valid in some cases by offering relevant resources for his context, even if repeated or already used by the user. As presented in the Subsection 4.4, the gathered results are directly related to the quality of the trails generated by the community of users. This means that improvements must be presented in the collection of the updated data, whether done by trail simulators or in fact by real users.

From the asymptotic analysis performed in section 5, with the use of dynamic programming, in specific bottom-up approach, we were able to improve the presented Vulcanus' algorithm. If a new asymptotic

analysis is performed on Vulcanus 2.0, the worst case scenario will still be the same Big O(n^3). However the worst case has a very unlikely occurrence. The changes performed aim to improve the average scenario. In average, the trails are composed of several elements (from 10 to 35), and only a few trails are similar and therefore relevant. With the use of relevance flag, summed up with bottom-up comparison, it was possible to in fact improve Vulcanus' algorithm.

6.2 Contributions

Table 10 presents a comparison of the works presented in Section 2 (Recommendation of Travel Plans for Elderly People [17], SOLVE-D [18], Project SAID [19], and User Modelling Wizard for People with Motor Disabilities [20]), including now the Vulcanus RS. This work addressed the Vulcanus' algorithm optimization and presenting its evaluation model that, through a hybrid approach (collaborative, context-based and knowledge-based) is able to offer resources to users with different needs in different contexts.

This hybrid approach is due to the use of trails generated by the community of users (collaborative), context information (context-based) and information of each of the registered resources (knowledge-based). Vulcanus 2.0 is still able to support both people with disabilities and elderly people, according to the convenience of the application using the RS. Vulcanus 2.0 uses concepts such as user profiles, contexts and similarity analysis, in the same way as the other presented works. The main difference and contribution of Vulcanus 2.0 to other works is the use of trails and not only history. While histories deal only with data related to the user's past, trails consider the chronological information as a whole, obtaining a context of use at each new resource insertion.

Table 10: Comparison between the selected works, including Vulcanus 2.0.

Attribute	**Travel Plans for Elderly People**	**SOLVE-D**	**Project SAID**	**User Modeling for People with Disabilities**	**Vulcanus**
Classification of the used RS	Hybrid (Collaborative, Content-based and Knowledge-based)	Context-based	Probability-based	Context-based	Hybrid (Collaborative, Content-based and Knowledge-based)
Support to users	Elderly	Disable and elderly	Elderly	Disabled and elderly	Disabled and elderly
Context use	No	Yes	Yes	Yes	Yes
User profile use	Yes	Yes	Yes	Yes	Yes
History or trail use	Only history	Only history	None	None	Trails
Similarity Analysis use	Yes	Yes	No	No	Yes

6.3 Future Works

Based on the obtained data and through the evaluation scenarios, we found questions yet to be better explored and improved. The first of them is related to the quality and fidelity of the community trails. Despite the efforts to obtain real trails, the presented trail simulator was unable to generated well-defined trails or trails with a specific purpose. It was able to generate feasible trails, but some of them do not have a well-determined logic sequence or a specified purpose, therefore questioning the trail's quality.

In Vulcanus 2.0 itself, it is possible to improve the *Similarity Module*, adding a function in order to avoid the excessive recommendation of resources already presented in the user's trail. Scenario 3 showed that some situations validate the recommendation of a resource already present in the user's trail. However, this recommendation must be performed with caution, only in situations of high similarity between trails, as the innovation in the resource supply may be affected. Finally, the acceptance of Vulcanus 2.0 may only be evaluated by real users. For this purpose, Vulcanus 2.0 must be integrated with an application, such as the prototype Hefestos [15]. It is through the evaluation of these aspects that we may obtain irrefutable and clear results. The results obtained in this work are important because they show aspects that need to evolve, but only from real implementation it will be possible to precisely validate Vulcanus 2.0.

References

[1] V. J. Breternitz and L. A. Silva, "Big data: Um novo conceito gerando oportunidades e desafios," *Revista Eletrônica de Tecnologia e Cultura*, vol. 2, no. 2, pp. 1–8, 2013.

[2] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*. Cambridge University Press (CUP), 2009. [Online]. Available: http://dx.doi.org/10.1017/CBO9780511763113

[3] M. Weiser, "The computer for the 21 st century," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 3, no. 3, pp. 3–11, jul 1999. [Online]. Available: http://dx.doi.org/10.1145/329124.329126

[4] G. C. Vanderheiden, "Ubiquitous accessibility, common technology core, and micro assistive technology," *ACM Trans. Access. Comput.*, vol. 1, no. 2, pp. 1–7, oct 2008. [Online]. Available: http://dx.doi.org/10.1145/1408760.1408764

[5] R. R. Oliveira, F. C. Noguez, C. A. Costa, J. L. Barbosa, and M. P. Prado, "SWTRACK: An intelligent model for cargo tracking based on off-the-shelf mobile devices," *Expert Systems with Applications*, vol. 40, no. 6, pp. 2023–2031, may 2013. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2012.10.021

[6] J. L. V. Barbosa, R. M. Hahn, D. N. F. Barbosa, and A. I. da Costa Zanel Saccol, "A ubiquitous learning model focused on learner interaction," *IJLT*, vol. 6, no. 1, p. 62, 2011. [Online]. Available: http://dx.doi.org/10.1504/IJLT.2011.040150

[7] H. D. Vianna and J. L. V. Barbosa, "A model for ubiquitous care of noncommunicable diseases," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 5, pp. 1597–1606, sep 2014. [Online]. Available: http://dx.doi.org/10.1109/JBHI.2013.2292860

[8] J. M. Silva, J. H. Rosa, J. L. V. Barbosa, D. N. F. Barbosa, and L. A. M. Palazzo, "Content distribution in trail-aware environments," *Journal of the Brazilian Computer Society*, vol. 16, no. 3, pp. 163–176, jul 2010. [Online]. Available: http://dx.doi.org/10.1007/s13173-010-0015-1

[9] I. G. Cardoso, B. Mota, J. L. V. Barbosa, and R. da Rosa Righi, "Vulcanus: A recommender system for accessibility based on trails," in *2015 Latin American Computing Conference, CLEI 2015, Arequipa, Peru, October 19-23, 2015*, 2015, pp. 1–12. [Online]. Available: http://dx.doi.org/10.1109/CLEI.2015.7360003

[10] T. Wiedemann, "Simcop: Um framework para analise de similaridade em sequencias de contextos," Thesis of Masters on Applied Computing, University of Vale do Rio dos Sinos, São Leopoldo, 2014. [Online]. Available: http://www.repositorio.jesuita.org.br/handle/UNISINOS/4755

[11] J. H. da Rosa, J. L. Barbosa, and G. D. Ribeiro, "ORACON: An adaptive model for context prediction," *Expert Systems with Applications*, vol. 45, pp. 56–70, mar 2016. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2015.09.016

[12] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, and W. V. de Velde, "Advanced interaction in context," in *Handheld and Ubiquitous Computing*. Springer Science Business Media, 1999, pp. 89–101. [Online]. Available: http://dx.doi.org/10.1007/3-540-48157-5_10

[13] A. K. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, feb 2001. [Online]. Available: http://dx.doi.org/10.1007/s007790170019

[14] A. Wagner, J. L. V. Barbosa, and D. N. F. Barbosa, "A model for profile management applied to ubiquitous learning environments," *Expert Systems with Applications*, vol. 41, no. 4, pp. 2023–2034, mar 2014. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2013.08.098

[15] J. Tavares, J. Barbosa, I. Cardoso, C. Costa, A. Yamin, and R. Real, "Hefestos: an intelligent system applied to ubiquitous accessibility," *Universal Access in the Information Society*, aug 2015. [Online]. Available: http://dx.doi.org/10.1007/s10209-015-0423-2

[16] Wheelmap.org, "Wheelmap.org website," 2015. [Online]. Available: http://wheelmap.org/en/

[17] B. Batouche, D. Nicolas, H. Ayed, and D. Khadraoui, "Recommendation of travelling plan for elderly people according to their abilities and preferences," in *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*. Institute of Electrical & Electronics Engineers (IEEE), nov 2012. [Online]. Available: http://dx.doi.org/10.1109/CASoN.2012.6412423

[18] M. Sohn, S. Jeong, and H. J. Lee, "Self-evolved ontology-based service personalization framework for disabled users in smart home environment," in *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. Institute of Electrical & Electronics Engineers (IEEE), jul 2013. [Online]. Available: http://dx.doi.org/10.1109/IMIS.2013.48

[19] I. Zinnikus, A. Bogdanovich, and R. Schäfer, "An ontology based recommendation system for elderly and disabled persons," in *Workshop Adaptability and User Modeling in Interactive Systems (ABIS 2002)*, 2002.

[20] W. Kurschl, M. Augstein, H. Stitz, P. Heumader, and C. Pointner, "A user modelling wizard for people with motor impairments," in *Proceedings of International Conference on Advances in Mobile Computing & Multimedia - MoMM13*. Association for Computing Machinery (ACM), 2013. [Online]. Available: http://dx.doi.org/10.1145/2536853.2536860

[21] E. Commission, "2013 access city award for disabled-friendly cities goes to berlin," 2012. [Online]. Available: http://europa.eu/rapid/press-release_IP-12-1309_en.htm

[22] A. Dey, G. Abowd, and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-Comp. Interaction*, vol. 16, no. 2, pp. 97–166, dec 2001. [Online]. Available: http://dx.doi.org/10.1207/S15327051HCI16234_02

[23] M. Satyanarayanan, "Pervasive computing: vision and challenges," *IEEE Pers. Commun.*, vol. 8, no. 4, pp. 10–17, 2001. [Online]. Available: http://dx.doi.org/10.1109/98.943998

[24] L. V. Toscani and P. A. Veloso, *Complexidade de algoritmos*. Porto Alegre–RS: ARTMED Editora SA, 2008.