Semantic Location Based Services for Smart Spaces

Kostas Kolomvatsos, Vassilis Papataxiarhis, Vassileios Tsetsos

Pervasive Computing Research Group, Communication Networks Laboratory,
Department of Informatics & Telecommunications, University of Athens,
Panepistimiopolis, Ilissia 15784, Greece.
{kostasks, vpap, b.tsetsos}@di.uoa.gr

Abstract. Enhancing the physical environment of users with IT and communication elements is one of the main objectives of the pervasive computing paradigm. The so-called "smart spaces", which are typical pervasive computing environments, combine computing infrastructure with intelligent and context-aware services in order to advance the users' computing experience. In this paper we describe the basic metadata-related infrastructure that is required for delivering semantics-aware location-based services in smart spaces. This infrastructure involves geometric and ontological spatial representation as well as graph- and knowledge-based navigation algorithms.

Keywords: navigation, location-based services, Semantic Web, ontology

1. Introduction

Nowadays, one can observe an integration of the technology with the environment, giving the opportunity to people to utilize it at every time and any place. This is the core idea in pervasive computing [3] and it deals with the distribution of computing devices, such as wearable computers, sensors and other computational elements, in the physical world. Through this highly distributed infrastructure, users are able to gain access to various information sources and services. This way, the physical environment is transformed to a "smart space". Among the main applications for "smart spaces" are the Location Based Services (LBS). LBSs are information services, usually accessible through mobile devices, that correlate spatial and non-spatial data in order to enable location-aware content delivery [1]. However, existing LBS systems have limitations in the management of dynamic location-dependent content as well as in the interoperability between different platforms and application domains [12]. These limitations are mainly due to the poor representation techniques that are adopted for spatial data and application content. Most existing systems rely on geometric spatial information, which is incapable of expressing location semantics and does not adhere to some standard (especially for indoor spaces, where GPS is not available).

Navigation services are the most challenging LBS due to their complexity. In a smart space, they should take into account users' physical and perceptual characteristics as well as the dynamic space semantics (e.g., temporary obstacles in the path elements). Both the path selection and guidance processes should reason over the current user context, in order to provide optimal user experience.

It is expected that Semantic Web technologies can provide solution to the aforementioned problems by providing means to resolve interoperability issues as well as to implement intelligent and personalized LBS. Based on the Semantic Web, a semantic navigation service would be based on a spatial ontology and on appropriate knowledge-based inference for path selection. In this work we present such a semantic navigation system that can help users, even with sensory, physical or mental disabilities, to navigate in an indoor environment. The proposed solution is also applicable to outdoor navigation scenarios. Moreover, the proposed representation techniques can also support LBSs other than navigation.

The present paper focuses on selected practical issues not covered in the literature (e.g., semi-automatic population of spatial ontologies). More details on the entire LBS platform can be found in [4]. In Section 2 we give a short description of the ontology used to define the spatial elements as well as a GIS-based methodology for the creation of the ontology instances. Section 3 describes the algorithms that can be supported by this approach and that have been implemented in our system. Some related work is surveyed in Section 4. The paper concludes with a brief discussion on some key research issues.

2. Spatial Modeling

2.1. Spatial Ontology

Our approach is based on a semantic description model that exploits concepts from a spatial ontology in order to describe basic elements of navigation paths. The ontology used is an extension of the Indoor Navigation Ontology (INO) presented in [4].

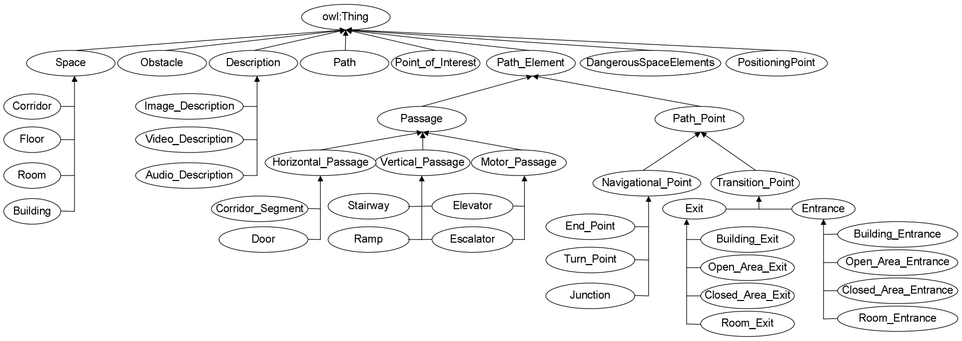

Figure 1. The main concept hierarchy of INO (arrows represent 'is-a' relationships)

As we can see in Figure 1, INO defines concepts that correspond to every basic spatial element that can be found in an indoor environment, and specifically in or near a path. Some key concepts are Space, Path, Path_Element, Path_Point, Corridor, Passage, Obstacle, Positioning_Point, Point_of_Interest, Entrance, etc. (their definition can be found in [4]). Additionally, INO includes several relationships among concepts in order to describe the connections between them. Many concepts are defined through necessary and sufficient conditions. This design decision considerably simplifies the population process (see Section 2.3). Specifically, the class of each passage is deduced from the class of its endpoints. Hence, during INO population we only have to manually classify the Path_Points (in the GIS, as described in the following section), while the passages are automatically classified by the reasoning engine.

Furthermore, INO contains concepts that can be aligned to user characteristics, typically defined in a user ontology. This way, we are able to provide personalized services based on INO. Finally, each spatial concept has an associated description type, information useful during guidance or for advanced path selection techniques (see also Section 3.4).

Reasoning involved in path searching and user guidance is based on instances extracted from files produced by a Geographic Information System (GIS), axioms holding for the INO relationships (e.g., transitive relationships) and rules (i.e., user profile). In the following sections we describe the two phases followed for INO population: a) appropriate annotation of GIS data, and b) automatic creation of the ontology instances from GIS data.

2.2. Ontology-driven map annotation with GIS

As described in the previous section, the INO ontology provides a number of concepts for spatial element representation. In our navigation system these concepts are used as a basis for instances creation. In order to define the navigation elements of an indoor environment, we use a GIS system. GIS systems help developers to visualize data concerning the location of various spatial elements such as entrances, navigational points (end points, turn points, junctions), etc. In our model, we categorize points according to the concepts defined in INO. Hence, we decided to use a layered architecture, where each layer corresponds to a basic concept in INO ontology. For example, there are layers that describe points of building entrances and exits, room entrances and exits, elevators, ramps, etc. Figure 2 depicts this layered representation.

The blueprints of each floor are located at the lower layer and are used as a reference for the rest of the layers. Based on each floor's map, corridors are designed in order to work as reference for the markup procedure of the spatial elements. Each corridor (segment) is represented by a line. Upon these lines, the various points are defined as well as their most important semantic features. These features concern the floor in which each point is located, its geometry (x and y coordinates), its identification number, its label and a short description. For inter-floor connections (i.e., vertical passages) special metadata are used for associating their endpoints.

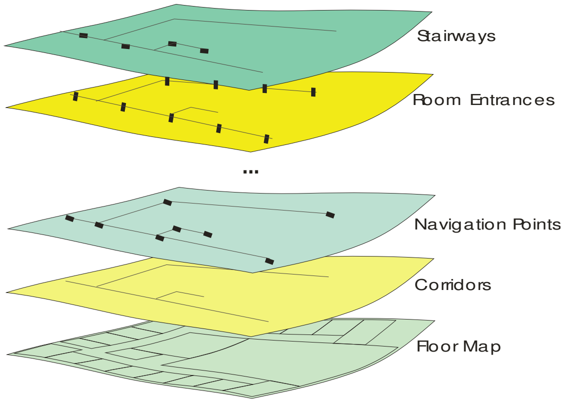

Figure 2. GIS Layers

2.3. GIS-based Ontology Population

Once the GIS data are in place, one can proceed with the actual ontology population. As already mentioned, each layer in GIS files corresponds to a spatial element class (or set of classes) defined in INO. Each of these layers is exported as a shape file which is further imported in a table of a spatial database. Subsequently, a series of algorithms are used to create the instances. First of all, we use an algorithm in order to decide for every point in a floor to which line it belongs to. The second step is to discover the endpoints of each line in order to help us in the next step (i.e., discovery of adjacent points). The rational behind this step is that an endpoint has only one neighbor, contrary to a point in the middle of the line, which has two neighbors. This procedure is repeated for all floors of the building, for all lines in a floor and for all points. The result is that for every point we store its floor, the lines in which it belongs to and its adjacent points in the floor. The final step is to create the individuals based on INO and the information that we have extracted from the previous steps. The outline of the described algorithm is presented in Listing 1.

2.4. Orientation Issues

An essential precondition for an indoor navigation service is the ability to determine, at any given time and for any given user moving along a route, her orientation. Orientation information includes data such as cardinal points and deviation from cardinal directions and, combined with positioning data, it can be translated into directional information, such as “left” or “right”. Utilizing orientation information in the context of indoor navigation services has not been widely explored yet, although it can advance the user experience through more intuitive routing instructions (e.g., “the third door at your left”, “the door opposite to the coffee machine”). The main obstacle encountered in orienting users in indoor spaces is the lack of efficient ways to directly

```
CreateIndividuals(INO, spatial DB tables)
Begin
  Create a list of the Floors
  Foreach Floor F
    Foreach line L in F
      Find the points that belong to L
      Find the endpoints of L
    Endfor
  Endfor
  Foreach Floor F
    Foreach line L in F
      Foreach point P in L
        If P is an endpoint then
          Find one adjacent point (based on coordinates)
        Else
          Find two adjacent points (based on coordinates)
        Endif
      Endfor
    Endfor
  Endfor
  Foreach Floor F
    Create INO Floor instances
    Foreach point P in F
      Create an instance in the INO class indicated by its GIS layer
    Endfor
    Foreach line L in F
      Create and INO Passage instance
      Perform reasoning (the instance is classified based on its
         endpoints and its definition in INO)
    Endfor
  Endfor
End
```

Listing 1. Algorithm for instance creation

acquire orientation information, since, unlike the outdoor navigation case, most built structures prohibit the use of GPS devices.

In order to resolve this issue, our approach uses a predetermined, arbitrarily chosen corridor segment as reference, with known deviation from the North. This corridor segment is explicitly represented both in INO and GIS. The angle between each corridor segment and the reference segment is calculated and the deviation from the North is computed, after adding the deviation of the reference segment to this angle. As a result, the segments can be classified, depending on their deviation from North, into one of eight cardinal directions (*north*, *northeast*, *northwest, east, south, southeast, southwest, west*). Whether a point is at the left or the right of the user, is decided by taking into account her orientation (e.g., through a digital compass).

3. Hybrid Navigation with Graphs and Rules

The INO instances, along with the ontological representation of user profiles and the content semantics, which are described in [4], can be used in a wide range of LBS/navigation algorithms. In order to exploit these metadata, a core part of these algorithms is implemented through rules. In the following sections some key design decisions and algorithmic issues for our system are discussed.

3.1. Why Rule-based Navigation?

A navigation algorithm can be considered as a special case of a path-searching algorithm applied to spatial graphs. Such algorithms typically try to calculate optimal paths with regard to some optimization criteria. For graphs representing spatial environments, the most common, and usually the only one, criterion is the Euclidian distance. There are many efficient algorithms for computing shortest paths, such as A* and optimized versions of the Dijkstra algorithm. However, when one introduces more than one criteria to a shortest path searching algorithm (e.g., quickest path, most popular path), its complexity increases dramatically. Specifically, we have a Multi-objective Shortest Path problem [5], which belongs to the broader category of multi-objective combinatorial optimization problems (known to be NP-hard [6]). Moreover, it is usually quite difficult to extend such an algorithm by adding more criteria. For example, let us assume that we have a spatial graph where each edge (i.e., corridor segment) is assigned a weight vector representing its physical edge and width. We also assume that we have designed an algorithm that calculates the shortest routes with edges exceeding a minimum width threshold. If we further want to add the height of each edge to its weight vector, the algorithm needs to be redesigned. However, this is not a trivial task, and even if we manage to design it, its computational complexity will increase considerably. Obviously, things become worse when we want to add more constraints to the path selection process.

Since in personalized human navigation systems we expect to have many different constraints during route calculation, we decided not to adopt such a "monolithic" approach. In particular, the complexity of the route instructions is a very important factor in human navigation, since people usually do not follow the shortest but the simplest path. The simplest-path algorithm proposed in [14] computes the "easiest-to-describe" path in a graph and has similar computational complexity with a shortest path algorithm. Although the resulting path is, in general, somewhat longer than the shortest one, it is simpler to be followed.

Hence, we have separated the path selection process to two discrete parts: a) identification of k-simplest paths, and b) application of constraints (in the form of simple, Horn-like rules) and preferences to these paths. The path that satisfies all or most of the constraints is guaranteed to be the *simplest acceptable path*. This modular approach has the following benefits:

- We take into consideration the route distance, which is usually the main criterion in human navigation.
- We can enforce as many constraints as we want and express them in a rather intuitive form of rules.
- The computational complexity does not scale exponentially with the addition of new route selection criteria. In fact, the MOSP problem is decomposed to a k Shortest Paths Problem and to the matching problem involved in rule triggering. The latter problem has been addressed since many years with efficient solutions even for large rule sets (in our case, navigation restrictions) [7]. Nowadays,

efficient algorithms have been implemented for both problems. Yen presented a generic solution of k Shortest Paths which can be implemented in O(kn(m+nlogn)) [16]. In recent years, RETE algorithms [15] have been proved efficiently in rule execution, facilitating the inference process.

The algorithmic and implementation details of this approach are provided in the following subsection. The capability of the approach to implement other novel navigation algorithms for smart spaces is demonstrated in Section 3.4.

3.2. The Navigation Algorithm

The implemented indoor navigation algorithm constitutes a hybrid rule-based approach of computing the “best traversable” path depending on the user profile, which is comprised of user abilities as well as preferences. In particular, the algorithm computes the optimal path with regard to a number of criteria, such as the total path length and the simplicity of route description.

The first step of the algorithm involves the creation of a “user-compatible” building graph, based on the user profile. This graph is obtained after applying a number of disability rules over the INO instances. These rules eliminate the path elements that cannot be traversed by a particular user. Once this restricted graph is calculated, the execution of the k-simplest paths algorithm follows.

Afterwards, several other rules assign bonus or penalty values to path elements, based on the profile of each user. In addition, the total cost of each path is computed as a function of the bonuses and the penalties of the path elements and the total length of the path. Finally, the simplest path is selected according to the total cost of the path computed in the previous step. The outline of the navigation algorithm is presented in Listing 2.

```
Navigate(INO, origin, destination, user profile)
Begin
  Create the building graph from INO
  Apply disability rules to INO instances
  Foreach Path Element PE
    If PE is accessible by the user
      Discover and remove the corresponding node and its edges from the
      building graph
    Endif
  Endfor
  Compute the k-Simplest Paths from origin to destination
  Foreach of the k-Simplest Paths
    Foreach Path Element PE
      Assign bonus/penalty value to PE, according to perceptual rules
      and user preferences
    Endfor
    Compute the total path length
    TotalPathRank = f(path length, bonus vector, penalty vector)
  Endfor
  Return the path with the maximum TotalPathRank
End
```

Listing 2. Outline of the navigation algorithm

3.3. Path Selection Example

In this section, we present a path selection example that demonstrates the execution results of the algorithm presented in the previous section. Figure 3 depicts the floor plan of a building. Initially, we assume that the user is located in point A and wants to move to point H. There are three possible loopless paths that she may follow. Examining these routes, it is established that the path ACFEH is the shortest path between A and H, but not the simplest one, due to the intersections it contains. On the other hand, the path ACFDGH is neither the shortest nor the simplest one. Specifically, it is the longer and the most complicated path between A and H and as a result it cannot be selected by the algorithm. Finally, the path ABDGH is the simplest since it contains only two turns (points B and G). Moreover, it is not much longer than the shortest one and, consequently, it is presented to the user as the "best traversable" path between A and H. For a more detailed analysis on the complexity of a path, one can refer to [14].

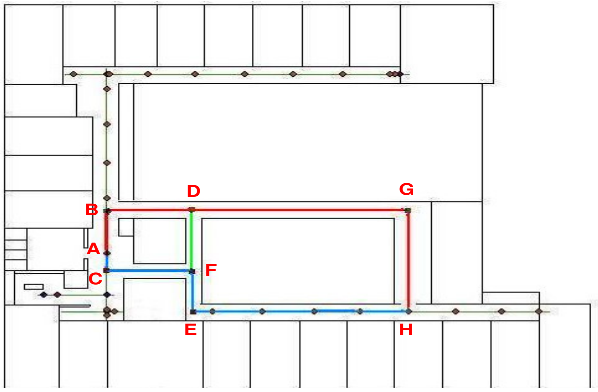

Figure 3. An example floor plan

3.4. Navigation Alternatives Supported by our Approach

Given that different users and different application domains have different requirements for navigation, new navigation algorithms should be devised. The following examples briefly introduce three algorithms, which we are currently experimenting with.

Example 1 - Content-based Navigation

A tourist is interested in ancient weapons and war exhibits. When she visits a large archaeological museum, she uses a tour guide that plans some routes of variable duration/length, all of which mostly containing exhibits of interest.

Example 2 – Presentation-based Navigation

A blind user needs some auditory assistance in order to find his way in a building. The building's spatial elements and points of interest are semantically annotated with descriptive images, texts, and/or sounds, which are used in the instructions provided by the system. In case the system knows that the user is blind, it can calculate paths that will be easier to describe to him (i.e., annotated with sounds and text, since the latter can be transformed to speech through appropriate software).

4. Related Work

Several researchers have recognized that semantics and metadata can improve the way location services work. For example, in [8] the authors present a Semantic Web framework for context-aware and location services. This framework, among others, specifies a context ontology which is used for annotating spaces and delivering advanced information services to pedestrian users.

Semantic spatial representation is at the heart of semantic LBS, since the quality of LBS algorithms depends heavily on the spatial modeling of the physical environment. As far as indoor navigation is concerned, only a few researchers have proposed practical, yet expressive, models. In our view, the most important one is presented in [9]. It is a hybrid model, which represents the space as semantic hierarchies of "locations" and "exits" that also carry geometric information (e.g., coordinates). More ontology-oriented approaches are presented in [10][11]. Even these models are necessary for developing LBS, we are not aware of any methodologies for (semi-) automatic instantiation of the respective models.

5. Conclusions and Discussion

Enhancing conventional context-aware services with semantics seems a promising solution for future pervasive computing environments. In this paper we have described some necessary elements and methods for implementing such services, with a special emphasis on navigation. We showed that implementing such services is feasible with the aid of Semantic Web technologies. However, there are some issues that deserve further investigation.

Firstly, existing rule engines are not fully integrated with ontology reasoners, thus complicating the inference process. Nowadays, although most prevalent rule languages share a common vocabulary with OWL (e.g. SWRL [13]), the lack of a single reasoning module handling both rule execution and ontology classification becomes clearer. Such a framework would facilitate the combination of ontologies with rules by solving possible inference conflicts between different engines.

Additionally, current rule engines do not fully or efficiently support advanced reasoning techniques, such as reasoning over concrete domains. Moreover, new challenges arise from the combination of monotonic (i.e., adopted by Semantic Web) with non-monotonic features (e.g., negation-as-failure). Finally, the application of rules to a model with a large number of instances seems to imply a significant time overhead, thus constituting a limiting factor for real-time applications.

References

1. Virrantaus, K., Markkula, J., Garmash, A., Terziyan, Y.V., ‘Developing GIS-Supported Location-Based Services’, In *Proc. of WGIS’2001 – First International Workshop on Web Geographical Information Systems.*, Kyoto, Japan. , pp 423–432, 2001.
2. Weiser, M. ‘The computer for the twenty-first century’, *Scientific American*, 265(3):94–104, Sept. 1991.
3. Satyanarayanan, M. “Pervasive computing: vision and challenges”, *IEEE Personal Communications*, Aug. 2001, Vol. 8, No. 4.
4. Tsetsos V., Anagnostopoulos C., Kikiras P., and Hadjiefthymiades S., "Semantically Enriched Navigation for Indoor Environments", *International Journal of Web and Grid Services (IJWGS)*, Vol.4, No.2,Inderscience Publ, December 2006.
5. Current, J., and Marsh, M. "Multiple transportation network design and routing problems: taxonomy and annotation", *European Journal of Operational Research*, 65, 4—19, 1993
6. Serafini, P. Some considerations about computational complexity for multi objective combinatorial problems. In J. Jahn and W. Krabs, editors, Recent advances and historical development of vector optimization, volume 294 of *Lecture Notes in Economics and Mathematical Systems*. Springer Verlag, Berlin, 1986.
7. Forgy, C. "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", *Artificial Intelligence*, 19, pp 17-37, 1982
8. Bikakis, A., Patkos, T., Antoniou, G., Papadopouli, M., and Plexousakis, D. “*A Semantic-based Framework for Context-aware Pedestrian Guiding Services*”, 2nd International Workshop on Semantic Web Technology For Ubiquitous and Mobile Applications (SWUMA), Riva del Garda, Trentino, Italy, August 2006
9. Hu, H., and Lee, D.L. (2004) ‘Semantic Location Modeling for Location Navigation in Mobile Environment’, *IEEE Mobile Data Management*, pp. 52-61.
10. Coenen, F. and Visser, P., ‘*A Core Ontology for Spatial Reasoning*’. Department of Computer Science, University of Liverpool, 1998.
11. Flury, T., Privat, G., Ramparany, F. ‘OWL-based location ontology for content-aware services’, *Proc. Artificial Intelligence in Mobile Systems*, Nottingham, pp52-58, 2004.
12. Kim, J. W., Kim, J. Y., Hwang, H. S., Park, S. S., Kim, C. S., Park, S. G. ‘*The Semantic Web Approach in Location Based Services*’, in ‘Computational Intelligence and its Applications’, ICCSA 2005, Singapore, Springer, pp 127-136.
13. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004b). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*, W3C Member Submission, 21 May 2004.http://www.w3.org/Submission/SWRL/.
14. Duckham, M. and Kulik, L., "Simplest Paths: Automated route selection for navigation," Accepted for COSIT'03, Lecture Notes in Computer Science, Springer-Verlag, 2003.
15. Forgy, C.L. 1982. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, 19:17–37.
16. J. Y. Yen, Finding the k shortest loopless paths in a network. *Management Science,* 17:712-716, 1971