

Topological World Modeling Using Semantic Spaces

Barry Brumitt & Steven Shafer

Microsoft Corporation, One Microsoft Way, Redmond, WA, 98053 USA
{barry, stevensh}@microsoft.com

Location models of the physical world play several roles in ubiquitous computing systems, including the provision of physically-scoped lookup, an abstraction layer between sensors and applications, and a shared metaphor with the user. Location models can be divided into two classes: metric and topological. Metric systems rely on distance measurements, while topological systems use relationships between abstract spaces, such as “near”, “contains”, etc. This paper describes a design for a topological space representation, discusses three ways in which such a model can be used, and poses five open problems in the ongoing development of semantic spaces representations.

Introduction

In ubiquitous computing⁴, the system aims to deliver services to people in the locations of their work, commerce, home life, and other activities. One of the key requirements for ubiquitous computing is the ability to determine where people are, what physical objects and software services are available at those locations, and how people can move from place to place. Some example questions that arise might be:

- Where is Joe?
- What display screens can Joe see?
- Which printer is closest for Joe to walk to?
- Is any sensor currently tracking Joe’s location?

One approach to answering such questions is to postulate the existence of a detailed floor plan of the space (building etc.), along with the location of all interesting devices in that space, and some method of dynamically tracking the locations of people in the space to a resolution of, say, one foot. With such a metric model of space, ordinary geometric calculations of distance, intersection, and path planning can be used to calculate the answers to these questions. Unfortunately, in real life, such detailed and exhaustive models are generally not available, and person-tracking at such a resolution requires a forbidding amount of hardware (e.g. a visual surveillance system¹ or a high-resolution active badge system³).

In the absence of such a metric model, it might seem that the above questions cannot be answered. However, a significantly simpler space model and tracking system that actually does suffice for answering these questions. Such a system might have a list of the rooms in the building, along with the key corridors and other spaces; a list of the doorways (etc.) that connect these places; and a list of the key objects and devices located in each named place. People’s locations could be determined by

asking them to take a specific action to assert where they are (such as touching a thumb-scanner when they enter a meeting room), or by using a low-resolution active badge system. Such a model might be called a topological rather than a metric model, because it represents containment and connectedness of spaces but not their size and shape; and it represents the fact of object containment in a space but not any specific geometric position of the object within that space. A similar distinction between metric and semantic spaces has been suggested by Pradhan². This paper presents *Semantic Spaces*, a characterization and database representation of spaces that is suitable to embody such a topological model of space.

Semantic Spaces Database

A semantic spaces database needs to be generally accessible to a large variety of applications. To simplify the development of such applications, minimizing the complexity of accessing the space information is a high priority. While ubiquitous computing systems are inherently distributed in nature, having a central database for obtaining location information keeps the application from needing to contact a huge set of smaller data stores which might only hold information about a small portion of the model. While centralized approaches can suffer from performance and reliability problems, modern relational database systems such as Microsoft SQLServer 2000 and Oracle9i are designed for high throughput and reliability. Beyond robustness and scalability, commercial databases also provide standard methods for data access and the ability to load custom code onto the server to perform more complex operations on the data, both of which further simplify application development. For these reasons, the Semantic Spaces database is currently implemented on SQLServer, and therefore, the following description of the logical schema will be given as standard relational database tables.

Spaces

A Semantic Spaces needs some representation for people, electronic devices, software services, and other things of interest, and should provide a single, uniform naming system for all of these. Semantic Spaces are not primarily concerned with these things, but rather with the spaces that contain them. So, they will be referred to as *atoms* and the name of an atom its *ID*. It is sufficient to assume that the surrounding system can determine all useful properties of an atom given the ID. For example, there might be a database that lists the IDs of all atoms and tells what type of atom they are and so forth.

In Semantic Spaces, the key element is a set of spaces, which have unique IDs, in a similar fashion to atoms as described above. They can be represented using a database table as follows:

Space ID	Friendly Name	Type

This denotes a table of entries, each of which has the above fields. The Space ID will be used as the system's "name" for the space. The Friendly Name allows the system to print messages about the space that will hopefully be meaningful to a person. A "Type" field allows description of the type of space. It is valuable to add this type indication about whether a space represents a room, a building, a work area inside a room, etc., to aid in limiting queries about spaces and atoms, and to otherwise differentiate types of spaces for query and UI purposes. However, the definition of these types remains a complex issue which will not be discussed here.

Containment

With Semantic Spaces, the goal is to reply to queries about world state, i.e. to search the model for the IDs of atoms and spaces that satisfy the conditions of a given query. One possibility would be to assume that the physical world is partitioned into unambiguous, non-overlapping pieces that exhaustively cover the area in question; then, each atom could be assigned to its piece. However, the real world is not so neat. For example, an office may be part of a floor, a wing, and a particular organizational group. Partitioning is, in effect, a way of centralizing all semantic boundaries into a single level of representation without any representation of aggregation or multi-level abstraction. Partitioning seems to be too heavy-handed and inflexible a tool to create a satisfying space representation.

What is needed, though, is the concept of one space being contained wholly within another, i.e. to be a subset or subspace of another. Mathematically, this can be expressed as $S \subseteq T$, and the equivalent proposition in terms of set membership, is $\forall p (p \in S) \Rightarrow (p \in T)$. In other words, the assertion $S \subseteq T$ means that, if we are given the assertion $p \in S$, we can then assume $p \in T$. Assertions of the form $S \subseteq T$ are represented in the database with the following table:

Space ID of superset (T)	Space ID of subset (S)

Note that this relationship defines a partial ordering on spaces, and thus implies a lattice structure among them. This is similar to a tree, except that each space could have multiple parents. Thus, one can still speak of the children and descendants of a space in the same way that one would speak of these concepts in a tree, except that the descendants (as well as the ancestors!) also form a lattice rather than a tree.

Presence

In space representation, a primary concern is what is inside a space, i.e. what devices, objects and people are located within it. The assertion that an atom a is within a space S is denoted simply as $a \in S$. A subtlety here is that S is not actually a collection of atoms, but a collection of points in the physical world; thus a actually represents the *location* of the atom. One consequence is that in this system, it is not possible to represent an object which is partly inside a space and partly outside of it. Assertions about the presence of atoms in spaces can be represented by a table as follows:

Space ID	Atom ID

Note that there is no assumption that an atom is only a member of a single space. Therefore, the assertion $a \in S$ by itself does not imply anything about the relationship between a and any other space. If combined with the assertion $S \subseteq T$, of course, it is possible to infer $a \in T$, which is very important for these semantics. This is the justification for propagating queries throughout the space lattice. For example, if the query is poised, “List the atoms in T ”, then the list must include the contents of T as well as the contents of all spaces in T ’s subspace lattice. Notice that this may not in fact produce the list of all atoms that are inside space T in the physical world; it merely produces the list of those that are *asserted to be* within T in the Semantic Space model of the world. Thus, all atoms produced are guaranteed to satisfy the query; however, there might in fact be additional interesting atoms in the world that satisfy the query but are not included in the list. Conversely, if we ask “List the spaces that contain a ”, then we would include every space that contains a directly, and all the superspace lattices of those spaces.

One additional issue that arises is the relationship between presence and containment. Formally, atoms a and spaces S, T are different *types* of things, so we say $a \in T$ but $S \subseteq T$. This means there are two separate tables – the Space Containment Table and the Atom Containment Table. However, if the Space IDs are drawn from the same set as Atom IDs, then presumably every space might also be represented as an atom of type “Space” and a single table could be used to represent containment of both atoms and subspaces.

Using Semantic Spaces

Presence of People

Because the locations of *people* are so important in ubiquitous computing, it is worth considering where the assertion might come from that a person is located within a particular space. Here are some possible origins for the assertion that Joe is in a particular room:

- Joe logs onto a computer that is known to be in this room.
- Joe uses an application on his Pocket PC to assert he is in this room.
- Joe speaks a recognition phrase to microphones known to be in this room.
- Joe touches a thumb scanner known to be in this room.
- Joe is wearing a badge which is tracked and interpreted to be in this room.
- Joe is presents his face to cameras known to be in this room
- Joe waves his Pocket PC to contact an IR beacon known to be in this room.

Through any of these technologies, the system might generate the assertion that Joe is in this room. In some cases, the system might automatically know when Joe leaves the room; in other cases, the system would know this only if Joe takes some action to inform it. If such an action is required, Joe might be likely to leave the room without doing this, in which case the assertion in the model would be incorrect. One could imagine augmenting the entry in the Atom Containment Table with a time stamp to indicate when the assertion was made, and possibly even adding a notation that indicates the source of the assertion, to help “expire” old assertions. Also, if Joe enters a location that is known not to intersect with this room, then the assertion that Joe is in this room can be expunged.

Presumably, similar technologies might be used to create assertions in the model about the present of objects and computing devices in various spaces.

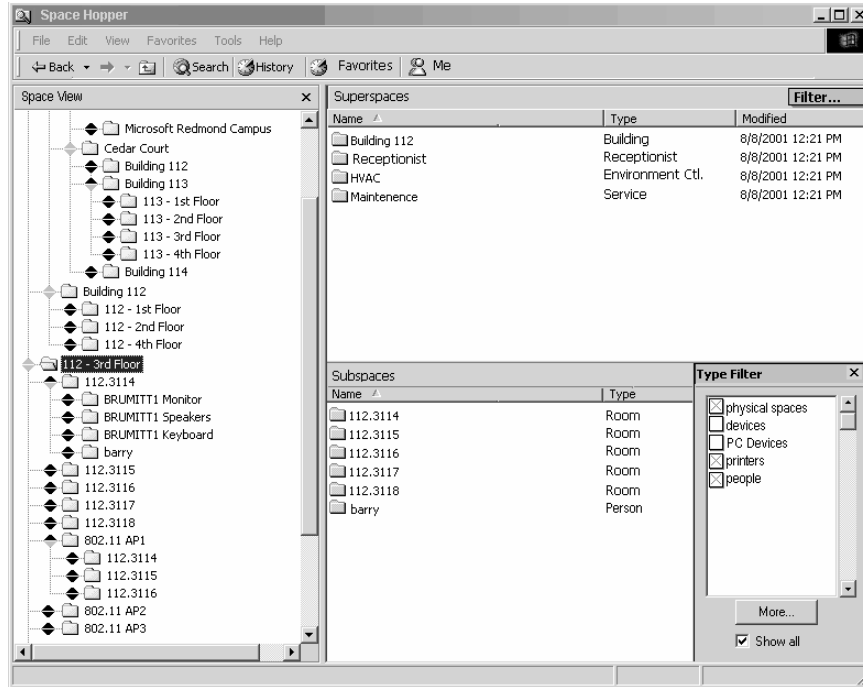
Geographic/Semantic Lookup

Many operations in ubiquitous computing scenarios require the ability to know the location of devices. For example, if the goal is to deliver an urgent message to a particular person, it is helpful to know if there are nearby displays, speakers, a PC, a cell phone, and so forth. Alternatively, if the task is to locate an appropriate printer, knowledge of the location of the user and the printers can be used to produce a list of printers which is limited to those nearby, rather than listing all printers available on the network – a potentially huge set!

Semantic spaces can be used for this sort of geometric lookup task. In the first example, the application responsible for messaging could look in the spaces which contain the person, and also in the spaces which are contained by the person to find potentially useful devices. If the application is aware of the ontology which describes spaces, it could use this knowledge to further limit the device set by not searching outside of the current room, hallway, or other immediate container.

Space Browsing Interface

For many applications, the goal of the semantic space representation is to aid the user in the selection of a device or service based on location information interpretable by the user. This is somewhat analogous to selection of files by navigation of a typical folder-based file explorer. The primary difficulty in designing such a space browser is that the space structure (a lattice) is much more difficult to visualize than the file structure (a tree). The following diagram shows one possible design for a space browser. [The folder icons are placeholders; in a real UI, they could be used to reflect type information]



The left pane of the window allows browsing of the topological structure, with arrows used to expand or collapse a node's parents or children. Note that, for example, since this structure is a lattice, "Building 112" appears twice in this view. The right-hand panes show the contents and containers of the selected space. The type filter window would allow filtering of that view by the selected type(s). The "Show All" option causes the pane to show **all** contained entities, including those anywhere in subspace lattice. This would make it easier to list all printers (for example) on a given floor, just by selecting the floor and filtering based on the type.

This type of interface could also be used to edit the space model, such as by adding new devices, spaces, relationships between spaces, and so forth. It is expected that a space model will be modified by many users, rather than being the product of a single person. One of the advantages to using containment as the primary relationship between spaces is that additions of new assertions about the world can never invalidate older ones.

Some open problems

The given semantic spaces description implies several interesting open problems. An incomplete list of such problems follows:

1. **Ontology** : What is the set of types that are used to describe spaces and atoms? Who controls this type definition, and how can it be extended?

2. **Efficient browsing** : UI's for lattice navigation are complex and hard to use. Is there a simpler interface for quick completion of typical tasks?
3. **Navigation** : By adding portals between spaces, and perhaps cost metrics, paths could be planned between spaces, so that the database could be used to answer questions like "How do I get to conference room 112/3912?"
4. **Access** : Who modifies the database? Who ensures it is correct?
5. **Person Tracking** : Different sensors have subtly different semantics; how can this information be successfully represented so that applications have the appropriate interpretation of the data ?

Conclusion

This paper introduces Semantic Spaces, a topological representation of the physical world for ubiquitous computing. The semantic space database has been implemented in a relational database system, using 3 logical tables to represent spaces themselves along with presence, and containment in those spaces. The database can be used for storing person (and device) location and for providing physically-scoped lookup. A sketch for a UI to help the user construct and navigate such spaces was also presented, though providing simpler interfaces remains an open challenge. Finally, a set of open problems related to semantic spaces was given.

References

1. Brumitt, B. L., Meyers, B., Krumm, J., Kern, A., Shafer, S. "EasyLiving: Technologies for Intelligent Environments", Handheld and Ubiquitous Computing, 2nd Intl. Symposium, , pp. 12-27, September 2000.
2. Pradhan, S. "Semantic Location", Personal Technologies, Springer-Verlag, Vol 4., No 4., pp. 213-217, 2000.
3. Ward, A. et al. "A New Location Technique for the Active Office", IEEE Personal Communications, Vol. 4, No. 5, pp. 42-47, Oct 1997.
4. Weiser, M. "Some computer science issues in ubiquitous computing", Communications of the ACM, 36(7):75--85, July 1993.