# On Location Models for Ubiquitous Computing

CHRISTIAN BECKER, FRANK DÜRR

*University of Stuttgart, Institute of Parallel and Distributed Systems (IPVS), Universitätsstraße 38, 70569 Stuttgart, Germany*

{Christian.Becker,Frank.Duerr}@informatik.uni-stuttgart.de

*Keywords: location model, ubiquitous computing, context-awareness, location-based services*

## Abstract

Common queries regarding information processing in ubiquitous computing are based on the location of physical objects. No matter if the next printer, next restaurant, or friend is searched for, a notion of distances between objects is required. A search for all objects in a certain geographic area requires the possibility to define spatial ranges and spatial inclusion of locations. In this paper we discuss general properties of symbolic and geometric coordinates. Based on that, we present an overview of existing location models allowing for position, range, and nearest neighbor queries. The location models are classified according to their suitability with respect to the query processing and the involved modeling effort along with other requirements. Besides an overview of existing location models and approaches the classification of location models with respect to application requirements can assist developers in their design decisions.

## Introduction

Location plays an important role in the domain of location-aware and context-aware systems. Especially in the ubiquitous computing domain location is commonly considered to be an important source of context [1] but not the only one [2]. However, whenever applications or users are interested in objects depending on their location or spatial relationship location models are required in order to provide notions about distances or ranges. This paper presents an overview of possible approaches, discusses existing work, and classifies the approaches and existing work according to their suitability to allow for range and nearest neighbor queries.

Information about locations is presented in different formats. Geometric coordinates as they are used by GPS refer to a point or geometric figure in a

Published in Personal and Ubiquitous Computing, 9(1), pp. 20-31, 2005.

multi-dimensional space, typically a plane or a three-dimensional space. The topological properties of such a space allow the calculation of distances between locations and their inclusion in other locations.
Symbolic coordinates on the other hand do not provide any reasoning about their spatial properties (distance and inclusion) without any additional information. Such coordinates are available via cell-ids in cellular networks, such as GSM or wireless LAN, as well as via other positioning technologies, such as radio frequency tags (RF ids) or infrared beacons.
Examples for the use of location information in applications are navigation services or location-based information systems, which select services based on their spatial proximity, e.g. the nearest printer, or notify when some events occur in the vicinity, e.g. a friend appears or an accident happens.
In order to allow such applications based on symbolic coordinates, a notion of spatial relations such as distance and inclusion is required. This information has to be modeled explicitly in a location model.
In this paper we will discuss general requirements on location management and derive three types of queries – position, nearest neighbor, and range - which should be supported by location models. The properties of symbolic coordinates are discussed in general. Based on these properties different kinds of location models are discussed and classified along their suitability to support the queries.

## System Model

Our system model consists of three kinds of components (cf. Figure 1):
*The location model* is the central part of our system model. It stores representations of static and mobile real world objects like representations of buildings and people, respectively. It is not the focus of this paper to describe how these objects are managed by an infrastructure, but we concentrate on the typical properties of the different kinds of location models. Examples of such location models are the Nexus platform [3, 4], the context information server [5], or the guide project [6].
*Applications* query the location model in order to carry out different tasks like navigation (see next section). They also update the location model, e.g. by inserting new objects into the model, deleting old objects, or by altering existing objects whose state has changed. For the context of this paper, we are interested in

the different kinds of queries and tasks that are carried out by these applications because they determine the internal structure and organization of a location model. As will be shown later in this paper, the suitability of a location model for distinct queries depends on its internal organization. This is especially of interest, when a location model is not tailored towards a single application or domain but should manage information for a variety of applications and their potentially diverging requirements.

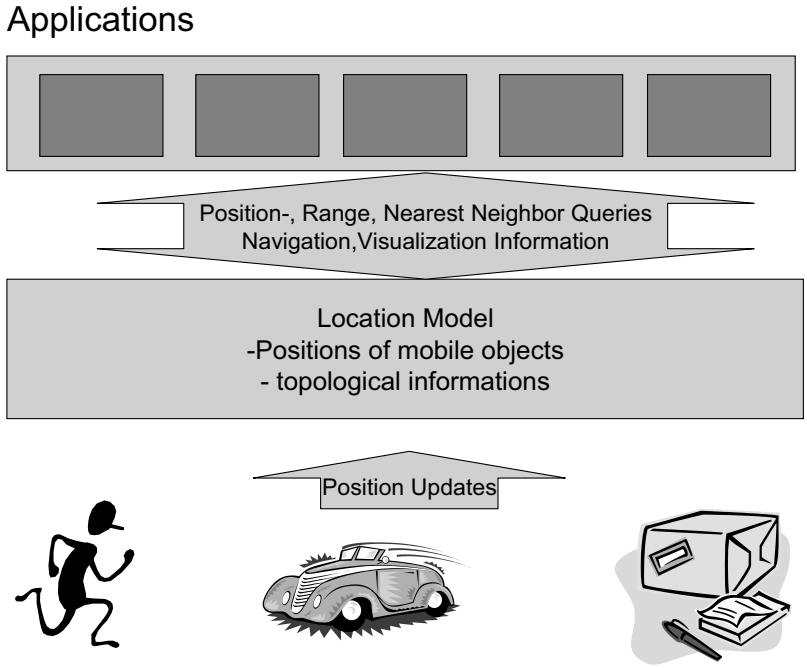


Figure 1: System model

*Positioning systems* update position information of mobile objects like persons or cars. The output of these systems also influences the location model as we will see in the next section. However, the multitude and variety of positioning systems and its discussion is beyond the scope of this paper. For the remaining part of this paper we will assume that a positioning system allows a mobile object or tracking system to issue a position update with a coordinate identifying a location to the location model. This is sufficient for the discussion of the properties of location models. However, the interested reader can find an overview of different positioning systems in [7]. Fusion aspects of different positioning systems into a common location framework are presented in [8]. In the following, a brief overview of the properties of coordinates as they are provided by current positioning systems is presented.

# Basic Properties of Coordinates

A *coordinate x* is an identifier which specifies the position of an object with respect to a given *coordinate system*. A coordinate system is a set *X* of coordinates. Some examples for different kinds of coordinates and coordinate systems are:

- Geographic coordinates in the WGS84, used by the GPS, are expressed as triples containing the geographic longitude, latitude, and the elevation above main sea level.
- The Active Bat System [9] is a high-resolution indoor positioning system providing three-dimensional coordinates – i.e. x, y, z value -- with respect to a local Cartesian reference system.
- The Active Badge System [10] provides symbolic identifiers for locations via infrared. Coordinates are the symbolic identifiers of the fixed IR sensors registering the users' active badges that transmit a unique identifier.

Two basic classes of coordinates can be identified from these examples: *geometric* and *symbolic coordinates*.

## Geometric Coordinates

Geometric coordinates define positions in the form of coordinate tuples relative to a reference coordinate system. We further distinguish global and local geometric coordinate systems. The World Geodetic System 1984 (WGS84) is a global reference system and thus can be used to define coordinates anywhere on this planet, whereas the Cartesian coordinates of the Active Bat System are typically only valid locally, e.g. in one room equipped with such a system.
Geometric coordinates can be used to calculate the distance between two geometrically defined positions. Through geometric operations it can also be determined if two areas overlap, touch each other, or one area contains the other, i.e. topological relations like spatial containment can be derived from the geometry of objects. Hence, geometric coordinates already allow simple spatial reasoning.

### Symbolic Coordinates

Symbolic coordinates define positions in form of abstract symbols, e.g. the sensor identifiers of the Active Badge system, or room and street names, etc. In contrast to geometric coordinates, the distance between two symbolic coordinates is not implicitly defined. Also topological relations like spatial containment cannot be determined without further information about the relationship between symbolic coordinates. Symbolic location models provide this additional information on symbolic coordinates.

## Requirements for Location Models

In order to derive requirements on location models and discuss their properties with respect to the organization, we will motivate queries to location models from the perspective of users and applications. Besides position queries, which are obviously needed in location-based applications, the necessity of nearest neighbor and range queries is motivated. This will serve as foundation of the later classification of location models. The choice of a distinct location model will dependent on the queries required by applications. Therefore, we have to consider these queries and tasks in order to assess the functional requirements for location models.

### Position Queries

The determination of the positions of mobile and static objects like users, buildings, bus stops, etc. is a common building block of location-based and context-aware systems. The tasks described below cannot be carried out without the known positions of objects. Therefore, all location models contain this information, but they differ in the way it is represented.
The definition of a position requires some form of *coordinates*. Based on an object's position actions can be carried out, such as teleporting the user's interface [9], controlling the input and output of applications to arbitrary spaces in the physical environment via projection techniques [11], or in industrial settings, such as a smart factory [12], the positions of resources and tools can be monitored in a production planning system. Such systems require a common interpretation of the coordinates in a specific global coordinate system. Within moving objects, such as

trains, local reference systems can help to address objects, such as travelers with respect to their compartment in the train and not their absolute position to the ground.

This shows that a general location model has to support *different coordinate reference systems, global and local* ones.

Beside well-known geometric coordinates, some positioning systems provide symbolic coordinates, e.g. the cell id in a cell-phone network or identifiers of infrared beacons, and often these symbolic coordinates can be interpreted more intuitively by users than geometric coordinates. Later we will show, how simple symbolic location models can be set up allowing for spatial reasoning with low modeling effort. Therefore, this kind of coordinates has to be supported as well.

## Nearest Neighbor Queries

A nearest neighbor query is the search for the *n* objects closest to a certain position. For instance, a user can search for the nearest restaurant with respect to his current position, or the next printer. Beside known object positions, the definition of a *distance function* on the coordinates is required for this type of queries. For geometric coordinates, the direct physical distance between two positions can be calculated using well-known formulas like Pythagoras in Cartesian systems. If only symbolic coordinates are modeled then the model must contain explicit definitions of distances between these coordinates, e.g. to define the distance between room number X and the printers in the rooms number Y and Z, since symbolic coordinates do not contain a natural embedment into a metric space.

There are other notions of distance that are often more relevant than the direct physical distance. For instance, for a pedestrian it might be impossible to cross a highway. Therefore, a restaurant across the highway with a direct physical distance of 100 m might be farther away than a restaurant with 200 m direct physical distance not located across this highway. In these cases additional model information like the road network a user uses to get from location A to B has to be taken into account. For such more complex nearest neighbor queries, this leads to similar requirements as for navigational tasks described in the next subsection, because “paths” between locations have to be found and their “lengths” have to be compared.

To sum up, a notion of "distance" is required in many context-aware or location-based systems. An explicit location model is required for symbolic coordinates as they do not provide implicit distance functions. Systems based on geometric coordinates can benefit from such a model as well, as spatial restrictions can be modeled, e.g. road networks.

## Navigation

Navigation systems become standard equipment in nowadays cars. Such systems require a location model to find paths between locations. Possible paths are defined by the transportation network (roads, train or bus routes, etc.) and consist of several interconnected locations. This means, it does not suffice to know the geometry e.g. of roads, but it is also important to know how to get from one location to neighboring locations, e.g. from one road segment to another road segment at a junction, and finally to the destination. Therefore, the *topological relation "connected to"* has to be modeled that describes these interconnections between neighboring locations (cf. Figure 2).

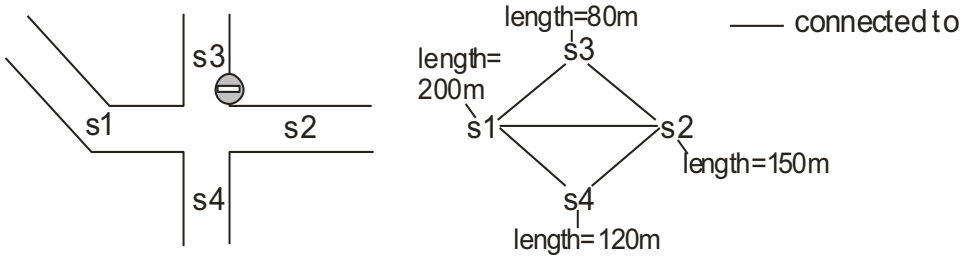


Figure 2: Road geometry (left) and road topology (right)

There are different kinds of navigational tasks, e.g. finding the shortest path or the fastest path. Finding for instance a suitable path for a person in a wheelchair requires additional information about locations, e.g. staircases or elevators. Therefore different attributes need to be modeled to implement these variants, e.g. the distance that has to be traveled to get from one location to another location, the maximum allowed speed on a road segment, the presence of stairs, which cannot be used with a wheelchair, etc. Even highly dynamic information like the current traffic situation on a road can be part of the model. In general, this means modeling some kind of weight on path segments. The "length" of a path is then calculated by summing up the weights of each path segment.

## Range Queries

A range query returns all objects within a certain geographic area. It can be used for instance to query the occupancy of a room as well as for a check whether an evacuation plan is processed correctly, i.e. if a room is empty before the fire doors are closed and sealed. Also, simple algorithms for new types of communication can be implemented on the basis of range queries, e.g. geocast [13], i.e. the sending of messages to receivers in a certain geographic area. First, a range query can be used to determine all receivers in the target area of the message. Secondly, the message is sent to these receivers, e.g. using multiple unicast messages.
First of all, object positions have to be known to answer a range query. Additionally, the *topological relation "contains"* has to be modeled, i.e. it has to be defined whether a coordinate lies within a spatial area. For geometric coordinates, this information can be derived from the known geometry. But for symbolic coordinates, this relation has to be defined explicitly. For instance, a model can define that the room 2.062 is on ("within") the second floor that in turn is part of ("within") a certain building, etc. Thus, querying for a larger area automatically includes all objects from locations that lie within that area.

## Visualization

Drawing maps is one of the most obvious application of location models. Maps can be used for many different tasks like positioning, navigation, etc., which we have already described in the subsections above. A map helps the user to execute these tasks manually or it is used to display the results of these tasks if they are carried out automatically. All model information introduced above can be visualized, but usually a map is drawn, which requires a more or less detailed *geometric representation* of these objects, depending on the desired level of detail (see below).

## Requirements

From the use cases presented above, the following requirements for location models can be derived. Note, that not all of these requirements have to be fulfilled at the same time. However, being aware of the application requirements is crucial in order to choose the appropriate location model organization.

Based on position, nearest neighbor, and range queries it can be concluded that a location model should provide:

- *Object positions:* Positions of objects have to be modeled in form of coordinates. Supported coordinates and reference systems are
    - *Geometric and symbolic coordinates*
    - *Multiple, local and global coordinate reference systems*
- *Distance function:* Distances between spatial objects have to be modeled. This can also be the "size" of a location, e.g. the length of a road segment, which represents the distance one has to travel when crossing this location in order to reach another location.
- *Topological relations:* The following topological relations between spatial objects have to be modeled:
    - *spatial containment* in order to allow range queries, and
    - *spatially connected to* for navigation services.

Furthermore, the position of objects alone is not sufficient for some applications which also require the direction of a moving object or the orientation of a user, e.g. in order to provide information about the building a tourist looks at.

- *Orientation:* In addition to positions of mobile objects, the orientation in the horizontal and/or vertical dimensions can be supported.

These requirements have to be regarded in conjunction with the requirement of *minimal modeling effort*. There are different factors that influence the modeling effort:

- *Accuracy:* The model should describe the real world as accurately as possible, i.e. the stored information should be consistent with the real world. Accuracy is not a question of the model type but of how the model is created and updated and of the dynamics of the modeled objects: Highly dynamic objects require high update rates, e.g. highly mobile objects will have to update their position frequently to get accurate position information. These issues are not the focus of this paper, and therefore accuracy will not be considered any further.
- *Level of detail:* The level of detail describes the precision or granularity of the model. Fine-grained models describe locations down to room level or below; coarse-grained models stop at buildings or larger. A flexible model allows both ends of the scale.

- *Scope:* The scope is the area covered by the model. Local models may only describe one single room, whereas global models at the other end of the scale describe locations all over the world.

The two last items are intimately connected. Highly detailed models usually only describe small parts of the world, because they require high modeling effort; coarse-grained models may have a larger scope [14]. Also the architecture used to manage the model plays an important role for the level of detail and scope. A federation of highly-detailed partial models with limited scope can be used to extend the scope of the (federated) model and make highly detailed global models feasible [4]. In this paper we do not consider how location models are management, but we concentrate on the general properties of the different kinds of location models. The following discussion first addresses location models for geometric and symbolic coordinates. Then the integration of geometric coordinates into symbolic location models leading to hybrid location models is discussed. Based on this discussion a classification of the general approaches is presented and existing work is classified.

## Geometric Location Models

Geometric models describe locations by geometric figures. If not only global coordinate systems are to be used but also local ones, the position and orientation of local systems with respect to other local systems or the global system has to be defined in order to translate coordinates of one system to other systems.
On the basis of geometric coordinates the topological relation “contained in” can be derived. In contrast to the containment relation, the ”connected to” relation modeling e.g. doors connecting rooms cannot be derived from location geometries. This relation has to be modeled explicitly. If this information is modeled, it can be used to improve the notion of distances, e.g. by incorporating the distance a user has to travel in contrast to the direct distance reflected by the underlying geometry. However, it is also reasonable for a geometric location model to store the spatial containment relation explicitly since geometric operations are costly.

# Symbolic Location Models

In this section we describe different types of symbolic location models and discuss their suitability for the different types of queries described in the requirements section of this paper. Set-based, hierarchical, and graph-based models are presented.

## Set-based Model

A set $L$ of symbolic coordinates forms the basis for the set-based approach. Locations comprising several symbolic coordinates are defined by sub-sets of the set $L$. As a simple example consider a building with several floors. The set $L$ consists of all room numbers of this building. The second floor as shown in Figure 3 can be modeled by the set $L_{floor2}$ = {2.002, 2.003, …, 2.067}. Further arbitrary locations may be defined, e.g. the locations $A$ = {2.002, 2.003} and $B$ = {2.003, 2.005} in Figure 3.

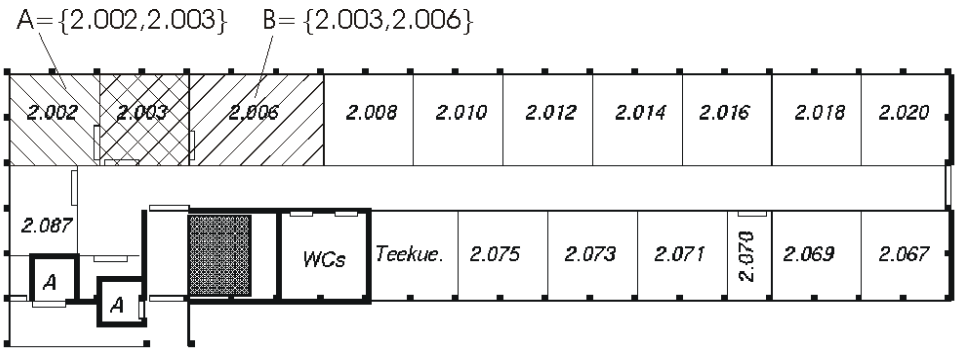


Figure 3: Set-based location model

This model can be used to determine overlapping locations and as a special case of overlapping locations the containment relation by calculating the intersection of two sets $L_1$ and $L_2$. If $L_1 \cap L_2 \neq \varnothing$, then $L_1$ and $L_2$ overlap. If $L_1 \cap L_2 = L_1$, then $L_2$ contains $L_1$. Thus, this model can be used for range queries where the range is defined by one set $R$ of symbolic coordinates, and all sub-sets of $R$ define locations within $R$.

This model can also be used to express a simple qualitative notion of distance between symbolic coordinates by modeling sets of "neighboring" symbolic coordinates, which we call neighborhoods (by $L_{con}$ we denote the set of neighborhoods). For instance the sets $A$ and $B$ in Figure 3 as well as the set $L_{floor2}$

defined above are such neighborhoods in $L_{con}$. Distances between the symbolic coordinates $x$, $y$ and $x$, $z$ are compared as follows:

$$\mathrm{d}(x,y) < \mathrm{d}(x,z) \Leftrightarrow \exists L_1 \forall L_2 \in L_{con}(x \in L_1 \wedge y \in L_1 \wedge x \in L_2 \wedge z \in L_2 \rightarrow L_1 \subset L_2)$$

That means, the two smallest neighborhoods containing $x$,$y$ and $x$,$z$, respectively, define the distance from $x$ to $y$ and $x$ to $z$. Consider for instance the three symbolic coordinates 2.002, 2.003, and 2.006. d(2.002, 2.003) < d(2.002, 2.006) because $A$ (the smallest neighborhood that contains 2.002 and 2.003) is a proper subset of $L_{floor}$ (the smallest neighborhood that contains 2.002 and 2.006 in our example).

To achieve a fine distance granularity, neighborhoods can be defined for each pair of directly connected locations, e.g. rooms which are connected by a door. For instance, the locations $A$ and $B$ introduced above are such locations. Larger neighborhoods are defined recursively by joining smaller neighborhoods which have non-empty intersections, e.g. the neighborhood C = $A \cup B$. By modeling pairs of connected locations, also possible paths can be derived. A negative effect of this approach is the huge number of resulting sets and the involved modeling effort.

Beside this qualitative notion of distance, this approach does not permit to define a quantitative notion of distance, e.g. to make statements like "the distance between $a$ and $b$ is as long as the distance between $c$ and $d$". Therefore, the support for queries related to spatial distances (e.g. nearest neighbor queries and navigation) is limited.

In contrast to set-based location models which do not contain explicit relations between locations, the following two models, i.e. hierarchical and graph-based, model relations between locations.

## Hierarchical Models

Hierarchical models consist of a set of locations $L$. The locations are ordered according to the spatial containment relation, i.e. a location $l_1$ is an ancestor of a location $l_2$ ($l_1 > l_2$), if $l_2$ is spatially contained in $l_1$. If locations do not overlap each other this leads to a tree-based model [15]. If overlapping locations are to be modeled the more general lattice-based model is applicable where intersections of locations are modeled by separate locations with more than one parent location [16,13]. Figure 4 shows an example of such a lattice-based model. The set of locations $L$ consists of the building $B$, the floors $F_1,\ldots, F_m$, two wings $W_1$ and $W_2$,

and several rooms $R_1,\ldots, R_n$. The locations $F_iW_j$ denote intersections of the floor $F_i$ and the wing $W_j$. Figure 4b also shows the relationship of the hierarchical models to the set-based approach. Locations in the hierarchy can also be interpreted as sets of symbolic coordinates. Overlapping locations are defined by the intersection of sets. Therefore, hierarchical models can be seen as a special case of set-based models.

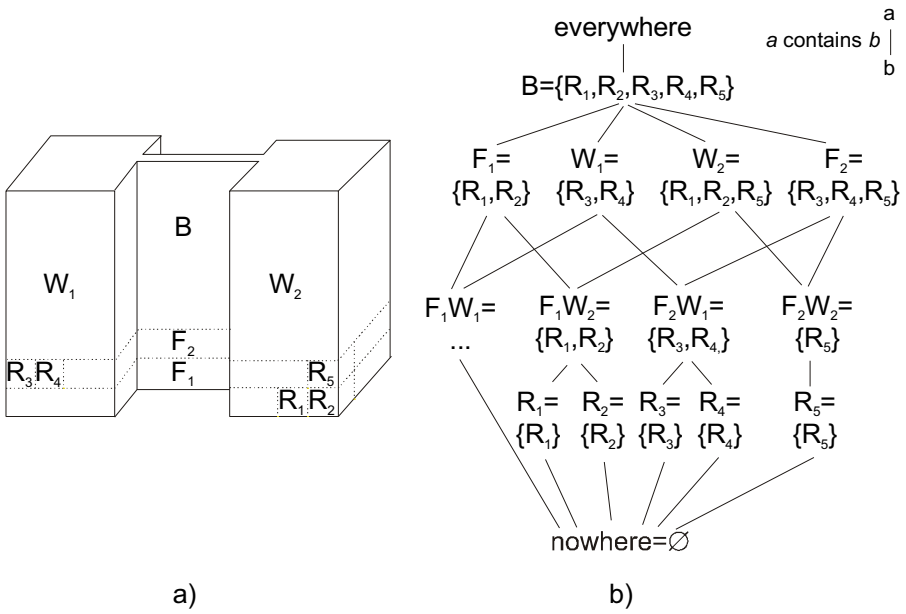


Figure 4: Hierarchical lattice-based location model

Because the hierarchical models are based on the containment relation they support range queries naturally. A range is defined by a location in the hierarchy and the descendants of this location denote locations within this range.
A simple notion of distance comparable to the one discussed in the previous sub-section can also be applied to hierarchical models:

Given three locations $l_1$, $l_2$, $l_3 \in L$. Then $d(l_1, l_2) < d(l_1, l_3)$, if $\sup(\{l_1, l_2\}) < \sup(\{l_1, l_3\})$.

$\sup(\{l_1, \ldots, l_n\})$ denotes the supremum (least upper bound) of a set of locations. For instance, the two rooms $R_1$ and $R_2$ located on the same floor and in the same wing in Figure 4 are considered to be closer to each other than the rooms $R_2$ and $R_5$, which are only in the same wing but on different floors ($F_1W_2 = \sup(\{R_1,R_2\}) < \sup(\{R_2,R_5\}) = W_2$). In some situations this interpretation of distance may be

counter-intuitive. If for instance a short connection exists between $R_2$ and $R_5$, e.g. stairs, the $R_2$ could be closer to $R_5$ than to some room located on the same floor and wing as $R_2$. Hierarchical models provide no means to model interconnections between locations, and therefore this situation can not be handled adequately. As for the set-based approach, this notion of distance is also only qualitative.

## Graph-based Model

In the graph-based approach, symbolic coordinates define the vertices $V$ of a graph $G = (V,E)$. An edge is added between two vertices if a direct connection between corresponding locations exists. Edges or vertices can be weighted to model distances between locations. Figure 5 shows an example of a graph-based model for the already presented second floor of a building. In this example the distance between two coordinates is just the number of hops but with additional information a higher accuracy could be achieved. [17] gives a deeper discussion of this aspect of graph-based models.

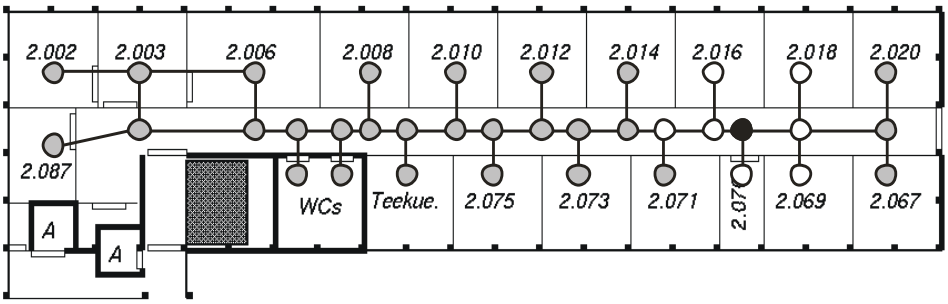


Figure 5: Graph-based model

From the construction of the graph it is already clear that a graph-based model supports the definition of the topological relation connected to as well as the explicit definition of distances between symbolic coordinates. It is therefore well-suited for nearest neighbor queries as well as navigation. For the latter the edges or nodes can be further attributed to model e.g. speed limits, vehicle restrictions, etc. [18].

For range queries first the range itself has to be defined, i.e. an area has to be described within which we want to search for included objects. The only locations which are explicitly defined in the graph-based model are the nodes of the graph,

e.g. the rooms shown in the example above. This is surely a very limited set of ranges. Because the graph-based model allows to define a distance between symbolic coordinates this distance can be used to define ranges. That means, an object is in the area, if the distance between its position and a reference location is at most the radius of the area. In Figure 5 for instance the white locations are within the range defined by a reference location marked black and the radius 2, thus all objects at these locations are within this range. What we are missing is the possibility to *explicitly* define bigger locations comprising several smaller locations, e.g. a whole floor, building, or even parts of a city. In the next section we will show how this limitation can be overcome by combining the different types of symbolic location models.

## Combination of Graph-based and Set-based Symbolic Models

Our discussion of the different location models has shown that for symbolic coordinates the graph-based approach supports queries based on distance and the definition of connected locations well, whereas the set-based approach can be used for range queries with explicitly defined locations like floors, building, etc. representing ranges. Therefore, a combination of graph-based and set-based symbolic locations models can be used to combine the benefits of both types of models.

The set-based part of the combined symbolic location model consists of a set of symbolic coordinates. Locations are sub-sets of this set of locations, e.g. representing rooms, floors, buildings, etc. This part of the model is used for range queries as described in the section about set-based models.

In the graph-based part of the combined model, locations are connected by edges if a connection between these locations exists in the real world. For instance, two rooms will be connected in the graph, if there is a door between these two rooms; two floors will be connected if stairs lead from one floor to the other, etc. As mentioned in the previous section, edges can also be weighted to model different distances. Figure 6 shows an example of the resulting combined model.

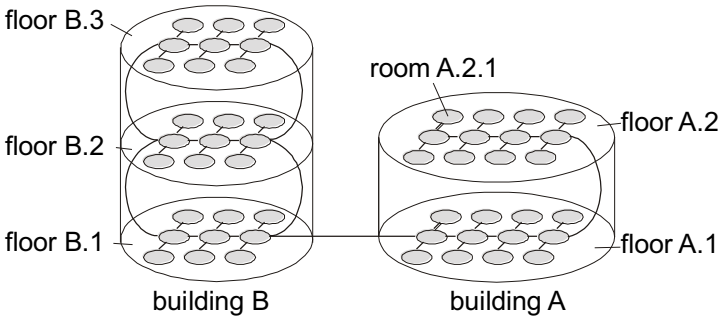


Figure 6: Combined symbolic location model

Besides the already mentioned support for different topological relations and distances and the range and nearest neighbor queries based on this information, this model shows another interesting feature. It allows to generate views with different levels of detail. Figure 7 shows three examples. The first example shows the rooms on one particular floor and their connections. This view will be used if a very fine granularity is required, e.g. if we are searching for the next printer. Figure 7b shows only the floors of building *A*. Floor A.1 and A.2 are connected because elements of Floor A.1 and A.2 have a connection – e.g. two hallways connected by an elevator. Finally, Figure 7c depicts only buildings and the paths between them. The latter could be used in a scenario where only coarse-grained location information suffices, and so it allows to generate small models that cover large areas, e.g. a whole city district.

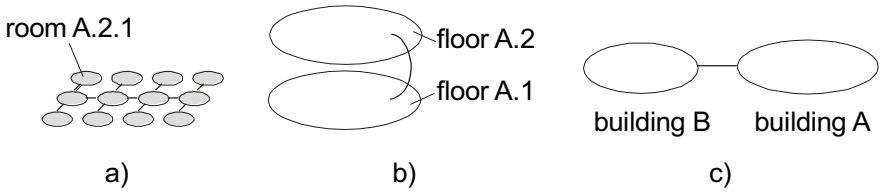


Figure 7: Levels of detail

## Summary

We now summarize the properties of the different types of symbolic location models presented in this section.

| symbolic model type | supported coordinate types | modeling effort[1] | distance support | "connected to" relation support | containment relation support |
|---|---|---|---|---|---|
| set-based | symbolic | high | limited | yes | good |
| hierarchical | symbolic | low to medium | very limited | no | good |
| graph-based | symbolic | low to medium | good to very good | yes | limited |
| combined (set-based & graph-based) | symbolic | medium | good to very good | yes | good |

Table 1: Properties of symbolic location models

We see that the graph-based approach as well as the hierarchical models support the containment relation well, making them suitable for range queries. The graph-based approach is well-suited for all kinds of queries where distance plays an important role, e.g. nearest neighbor queries and navigation. The combined symbolic location model combines the benefits of all other symbolic model types at the cost of higher modeling effort.

Still the accuracy of the combined model can be further improved by adding geometric information. The next section presents different hybrid models, which integrate symbolic and geometric information.

## Hybrid Location Models

The combined symbolic location model presented in the previous section shows how the benefits of set-based and graph-based models can be integrated into a common symbolic model. There are two major arguments for additionally adding geometric information to such a symbolic model. First, geometric information can be used to achieve higher accuracy and precision for all kinds of distance related queries. Secondly, arbitrary geometric figures can be used for instance to define

[1] Modeling effort is always dependent on the granularity and scope of location information as stated in the requirements section. Therefore, we give a range here.

ranges for nearest neighbor queries, whereas symbolically defined locations are always restricted to a given structure.
We distinguish between two types of hybrid location models. The first approach, which we call the sub-space approach, stores geometric information for every modeled location. The second approach only stores geometric information for some locations, leading to partial subspaces.

## Subspaces

The basis for this hybrid location model is a symbolic model like the combined symbolic model presented in the previous section. Additionally, the geometric extent of locations is stored in the location model. The geometric extent can be either defined using a global reference system like the WGS84 or local reference systems where coordinates are only valid within a certain scope, e.g. in one building or room. Subspaces are formed by embedding coordinate systems into other coordinate systems by defining the position and orientation of embedded systems (a detailed description of this embedding of subspaces can be found in [15]). With this information, coordinates can be translated from one system to other systems, and thus coordinates of different systems can be compared.

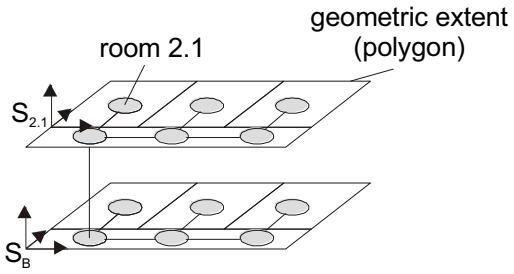

Figure 8: Hybrid location model with subspaces

Figure 8 shows a simple example of a hybrid location model using subspaces. The symbolic part of this model is based in a graph defining the interconnections between the rooms on a certain floor. The extent of every room is also modeled geometrically using the coordinate system $S_B$ of the building B. Within room 2.1 a local coordinate system $S_{2.1}$ is defined that is embedded into the system of building B. The system of building B in turn may be embedded into a global coordinate system. The known geometry can be used to define precise distances between rooms.

## Partial Subspaces

In contrast to the subspaces approach, the partial subspaces approach does not assume that the geometric extent for every location is modeled, but only for some locations. Figure 9 shows an example, where a geometric location model exists for the outdoor domain, but within buildings symbolic models are used. By linking geometric information to symbolic locations, the symbolic building models can be embedded into the global geometric model. The benefit of this integration becomes clear when we consider a range query with a geometrically defined range, e.g. a polygon drawn on a city plan. Users within a building may only know a symbolic position like room 2.1 in building B. Through the known geometric extent of the building, the user's position can be approximated geometrically with the geometry of the whole building. This approximated geometric position can be compared to the geometrically defined range of the query, and thus the query can be answered. Of course approximation has its limitations. For instance, using geometric areas within a building that is only modeled symbolically makes no sense. But it remains an interesting alternative that can be used to reduce modeling effort.

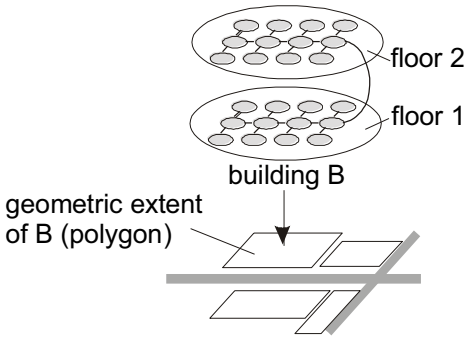


Figure 9: Hybrid location model using partial subspaces

## Discussion

A summary of the properties of the presented location models is shown in Table 2. In contrast to the purely symbolic models presented in the previous section, all hybrid models support geometric coordinates as well as symbolic coordinates. By using geometric information, distances can be modeled more accurately and precisely.

The spatial containment relationship does not need to be modeled manually if the geometry of locations is known. This information can be derived by using geometric operations. Still it makes sense to have a model that stores the containment relation explicitly to allow for efficient queries.
Geometric information can also be used to find out whether two locations lie next to each other, but connections like doors or junctions can no be derived from geometric information and therefore have to be modeled explicitly as for the symbolic approaches.
Compared to the subspaces approach the modeling effort can be reduced by using a partial subspace model where not every location is modeled geometrically. Still a geometry can be associated with location by using approximation.

| **model type** | **supported coordinate types** | **modeling effort** | **distance support** | **"connected to" relation support** | **containment relation support** |
|---|---|---|---|---|---|
| subspaces | symbolic, geometric | high to very high | very good | yes (if modeled explicitly) | yes |
| partial subspaces | symbolic, geometric | high | good to very good | yes (if modeled explicitly) | yes |

Table 2: Properties of hybrid location models

# Summary and Classification of Existing Approaches

This section briefly summarizes the properties of the different location models presented so far. Existing work is classified along the classes of location models. Table 3 summarizes the classes of location models, their properties and the existing work.
Since the discussion so far has shown that there is no location model serving all requirements at a time with similar modeling effort, designers of location management systems have to chose an appropriate structure of the underlying location model. Especially, the trade-off between supported queries and the involved complexity of the location models has to be taken into consideration.

| | supported coordinates | | supported queries | | | modeling effort | projects |
|---|---|---|---|---|---|---|---|
| | sym | geom | P | R | N | | |
| set-based | yes | no | good | good | basic | high | • Guide [6]<br>• comMotion [19]<br>• QoSDREAM [20]<br>• ActiveBadge [10]<br>• Open Distributed Office [21] |
| graph-based | yes | no | good | basic[2] | good | medium | • Aware Home [22]<br>• MavHome [23] |
| hierarchical | yes | no | good | good | basic | medium | • MOOsburg location model [24]<br>• Semantic Spaces [25] |
| combined symbolic | yes | no | good | good | good | high | • Active Map [1] |
| subspaces (hybrid model) | yes | yes | good | good | good[3] | very high | • Jiang [15]<br>• Leonhardt [26] |
| partial subspaces (hybrid model) | yes | yes | good | good | good | high | • Nexus [27,13]<br>• Semantic Location Model [28] |

**Table 3: Properties of location models and overview of existing implementations**

Table 3 classifies the location models with respect to the supported coordinate types (sym=symbolic, geom=geometric), the supported queries (P=position, R=range, N=nearest neighbor), and the modeling effort. Examples for projects

[2] "Range" defined by distance to reference location.

[3] If the "connected to" relation is modeled.

using the location model class are listed as well and are discussed in the following sub-sections.

## Set-based Location Models

Modeling symbolic locations as identifiers and mapping object ids to location ids in location services has been widely adopted. The Guide project identifies the locations of interest to tourists by the WaveLAN access point id [6]. The Active Badge system stores the identifier of a user's badge with the symbolic location where the badge has been observed. Without defining further locations as ranges only position queries can be processed with minimum modeling effort. However, an extension of such systems allowing for overlapping sets of locations and thus range queries has been used in the Open Distributed Office projects [21]. The modeling effort increases with the number of locations introduced to the system. QoSDREAM [20] relies on a mapping of location identifiers and object ids. By applying observers to sets of locations, applications can be notified when a mobile object has been observed in a set of locations. This provides means for range queries but causes considerable effort, since the overlay of observers modeling spatial inclusion has to be set up based on the basic sets.

## Graph-based Location Models

This class of location models naturally provides means to model distance making them suitable for all navigation oriented tasks. Applications can be found in the domain of smart environments [22,23]. Spatially scoped areas are modeled by the location users populate, e.g. floors and rooms, and a connection model defines connectivity and distance. Navigation services incorporating the positions of individual objects can be implemented that way. There is no direct notion of ranges. Either a combined approach is taken modeling ranges as an overlay structure – in the simplest case ranges are specified as sets of locations themselves – or ranges can be defined based on their extension, i.e. by a reference location and the distance to this location.

## Hierarchical Location Models

In contrast to graph-based models, which reflect distance well but require additional overhead to express ranges, hierarchical models are designed to reflect

the inclusion of locations. This allows to structure locations into a hierarchy. It is noteworthy that although approaches such as EasyLiving [25] or MOOsburg [24] only model the spatial inclusion between locations other kinds of hierarchical relations can be modeled such as an organizational structure. A company may structure its location into development, marketing, research, and production. A distributed systems development team – and its offices – may be organized to be nearer to the distributed systems research team in a hierarchy than the theoretical computer science research group in the offices nearby.

Ranges and their relations – spatially or with respect to other criteria such as organizational relations – are well reflected in a hierarchy. Distances do not come with a direct concept in such location models. One way to use a hierarchy to compare distances between positions is to consider the smallest locations in the hierarchy that contain these positions. That means, positions grouped by smaller locations are considered to be closer to each other than positions grouped by larger locations, e.g. two rooms on the same floor can be said to be closer than two rooms where the smallest common range is the building.

## Combined Symbolic Location Models

An obvious approach combining the benefits of graph-based and hierarchical location models are combined symbolic location models, such as those used in the Active Map [1]. Either a common data structure is applied that allows to reflect the inclusion relation as well as the connected-to relation between locations such as in [27], or two different location models are maintained where one reflects the distances and the other the ranges. Clearly, the expressiveness of such models combines the benefits of both models but with a trade-off with respect to the modeling effort, which basically consists of the effort of creating two location models. This effort is only justified when applications require range *and* nearest neighbor queries. This will likely be the case when a location model is set up to serve a number of applications, e.g. by providing an application spanning context model.

### Hybrid Location Models

Hybrid location models provide information about locations based on symbolic *and* geometric coordinates, which are used to define the spatial extent of locations.

Basically, all of the symbolic models above can be extended to a hybrid model by annotating location with their spatial extent. A graph-based model may use this information to calculate the weight of connections or rooms in order to provide more accurate distances. Since the effort of obtaining spatial extensions of locations is rather high, some projects consider a combined model as basis, e.g. [27] and [28]. The effort of annotating all locations in a location model with geometric information can be used to map the symbolic coordinates into a global, geometric reference systems realizing a subspaces approach [15]. If this is not necessary, a partial subspace approach can be taken. Such approaches can be realized either top-down or bottom-up. In [13] a top-down approach is taken that allows approximating the spatial extents of children in a location hierarchy by the extents of their father nodes. A bottom-up approach would annotate the leafs in a location model and approximate the spatial extents of a father node by the extents of its child nodes. The top-down approach allows the integration of an area that is modeled by a hierarchy of symbolic locations into a geometric model. The root of the integrated hierarchy is exact with respect to the annotated spatial extent whereas the approximation leads to some errors in the spatial extents along the hierarchy. In contrast to that, the bottom-up approach provides the highest accuracy at the leafs. The modeling effort is great for this approach if the hierarchy has many leaves. Clearly, it is application dependent which approach should be taken under given requirements.

## Conclusion

Modeling locations is crucial for most location-based or context-aware applications. Location models provide means for spatial reasoning based on coordinates, e.g. the determination whether a coordinate is within a given range or which coordinates are nearby. Although geometric coordinates already provide an implicit notion of distance and ranges, location models allow to model the constraints of the physical world, e.g. road networks or floor plans. For symbolic

coordinates like room or floor numbers, a location model with explicitly modeled relations between locations is essential to support queries beyond simple position queries.

The requirements of applications can be manifold. Since the structure of a location model determines which kinds of spatial reasoning can be processed, a number of location models may be appropriate. Beside the relevant queries a location model has to support, especially the modeling effort has to be taken into consideration when choosing a location model for an application or a platform serving a number of applications. A hybrid model managing geometric and symbolic coordinates supports all kinds of location-based queries very well but is at the same time the most complex type of location model. Location models managing only symbolic locations can be set up more easily. If, beside object positions, distance is the only relevant information, a graph-based symbolic model can be used, whereas range queries are supported very well by hierarchical symbolic models. If higher accuracy is required only partially within limited areas, a partial subspaces model, which augments a symbolic model partially with geometric information, might be the right choice.

The discussion of location models in this paper shows that there is no location model which satisfies all identified requirements at a time with a low modeling effort. Designers of context-aware applications and systems thus have to choose location models carefully with respect to the required spatial reasoning and the involved modeling effort.

# References


[1] William Noah Schilit: *A System Architecture for Context-Aware Mobile Computing*; PhD thesis, Columbia University, 1995

[2] A. Schmidt, M. Beigl, and H.-W. Gellersen: *There is more to context than location*; Computers and Graphics 23(6), 1999, 893-901

[3] F. Hohl, U. Kubach, A. Leonhardi, K. Rothermel, and M. Schwehm: Next Century Challenges: NEXUS – An open global infrastructure for spatial-aware applications; Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99), 1999, 249-255

[4] D. Nicklas, M. Großmann, T. Schwarz, and S. Volz: *A Model-Based, Open Architecture for Mobile, Spatially Aware Applications*; In Proceedings of the 7th International Symposium on Spatial and Temporal Databases (SSTD 2001), Redondo Beach, CA, USA, Jul 2001

[5] Glenn Judd and Peter Steenkiste: *Providing Contextual Information to Pervasive Computing Applications;* Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom), 2003, 133-142

[6] K. Cheverst, N. Davies, K. Mitchell, and A. Friday: *Experiences of developing and deploying a context-aware tourist guide: the GUIDE project*; In Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, Boston, Massachusetts, 2000, 20-31

[7] Jeffrey Hightower and Gaetano Borriello: *Location systems for ubiquitous computing*; Computer 34(8), 2001, 57-66

[8] J. Hightower, B. Brumitt, and G. Borriello; The Location Stack: A Layered Model for Location in Ubiquitous Computing; In Proceedings of the 4th IEEE Workshop on Mobile Computing Systems & Applications (WMCSA 2002), Callicoon, NY, 2002, 22-28

[9] A. Ward, A. Jones, and A. Hopper: *A new location technique for the active office*; IEEE Personal Communications 4(5), 1997, 42-47

[10] R. Want, A. Hopper, and J.Gibbons: The Active Badge Location System; ACM Transactions on Information Systems 10, 1992, 91-102

[11] Gopal Pingali, Claudio Pinhanez, Anthony Levas, Rick Kjeldsen, Mark Podlaseck, Han Chen, Noi Sukaviriya: *Steerable Interfaces for Pervasive Computing Spaces;* Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom), 2003, 315 322

[12] M. Bauer, L. Jendoubi, K. Rothermel, E. Westkämper: *Grundlagen ubiquitärer Systeme und deren Anwendung in der Smart Factory*; Industrie Management – Zeitschrift für industrielle Geschäftsprozesse, 19(6), 2003

[13] Frank Dürr, Kurt Rothermel: On a Location Model for Fine-Grained Geocast. In Proceedings of the Fifth International Conference on Ubiquitous Computing (UbiComp 2003), Seattle, WA, Oct 2003, 18-35

[14] K. Rothermel, D. Dudkowski, F. Dürr, M. Bauer, and Christian Becker: *Ubiquitous Computing – More than Computing Anytime Anyplace*; In Proceedings of the 49th Photogrammetric Week, Stuttgart, Germany, Sep 2003

[15] Changhao Jiang and Peter Steenkiste: *A hybrid location model with a computable location identifier for ubiquitous computing*; Proceedings of the Fourth International Conference on Ubiquitous Computing (UbiComp 2002), 2002, 246-263

[16] Wolfgang Kainz, Max J. Egenhofer, and Ian Greasley: *Modeling spatial relations and operations with partially ordered sets*; International Journal of Geographic Information Systems 7(3), 1993, 215-229

[17] T. Drosdol: *Unterstützung symbolischer Koordinaten im Lokationsmanagement*; Diploma thesis, University of Stuttgart, 2003

[18] Steffen Volz, Matthias Großmann, Nicola Hönle, Daniela Nicklas, and Thomas Schwarz: Integration mehrfach repräsentierter Straßendaten für eine föderierte Navigation; it+ti, Informationstechnik und Technische Informatik 5, 2002

[19] N. Marmasse, C. Schmandt: *Location Modeling*; In Proceedings of the Workshop on Location Modeling for Ubiquitous Computing, Atlanta, Georgia, USA, Sep 2001

[20] H. Naguib, G. Coulouris: *Location Information Management*; In Proceedings of the 3rd international conference on Ubiquitous Computing (UbiComp 2001), Atlanta, Georgia, USA, Sep 2001

[21] M. Rizzo, P. Linington, and I. Utting: *Integration of Location Services in the Open Distributed Office*; Technical Report 12/94, University of Kent, Canterbury, UK, 1994

[22] T. O'Connel, P. Jensen, A. Dey, and G. Abowd: *Location in the Aware Home*; In Proceedings of the Workshop on Location Modeling for Ubiquitous Computing, Atlanta, Georgia, USA, Sep 2001

[23] A. Roy, S. K. D. Bhaumik, A. Bhattacharya, K. Basu, D. J. Cook, and S. K. Das: *Location Aware Resource Management in Smart Homes*; In Proceeding of the First IEEE International Conference on Pervasive Computing and Communications (PerCom '03), Fort Worth, USA, Mar 2003

[24] Craig H. Ganoe, Wendy A. Schafer, Ulmer Farooq, and John M. Carroll: *An Analysis of Location Models for MOOsburg*; In Proceedings of the Workshop on Location Modeling for Ubiquitous Computing, Atlanta, Georgia, USA, Sep 2001
[25] Barry Brumitt and Steven Shafer: *Topological World Modeling Using Semantic Spaces*; In Proceedings of the Workshop on Location Modeling for Ubiquitous Computing, Atlanta, Georgia, USA, Sep 2001
[26] Ulf Leonhardt: *Supporting location-awareness in open distributed systems*; PhD thesis, Imperial College London, Department of Computing, 1998
[27] Martin Bauer, Christian Becker, and Kurt Rothermel: *Location models from the perspective of context-aware applications and mobile ad hoc networks*; Proceedings of the Workshop on Location Modeling for Ubiquitous Computing, UbiComp 2001, 2001, 35-40
[28] H. Hu and D. L. Lee: *Semantic Location Modeling for Location Navigation in Mobile Environments*; In Proceeding of the IEEE International Conference on Mobile Data Management, Berkeley, California, USA, Jan 2004

# Acknowledgements

*The second author gratefully acknowledges the financial support by the Deutsche Forschungsgemeinschaft (DFG) within the Center of Excellence 627 "Spatial World Models for Mobile Context-Aware Systems".*