Journal of Ambient Intelligence and Smart Environments 6 (2014) 71–91
DOI 10.3233/AIS-130238
IOS Press

An advanced location-aware physical annotation system: From models to implementation

Ahmad A. Alzahrani [a,b,*], Seng W. Loke [a] and Hongen Lu [a]
[a] *Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, Australia*
E-mail: aaalzahrani@students.latrobe.edu.au, {s.loke,H.Lu}@latrobe.edu.au
[b] *Department of Computer Science, King Abdulaziz University, Saudi Arabia*

Abstract. In the last few years, with the support of new communications and mobile technologies, Physical Annotations (PAs) have become popular in context-aware systems and pervasive computing. PA aims to enrich physical entities around us with useful information. Users can interact with inanimate objects, annotate them and share annotations with others. It helps daily life in different aspects, such as tourism, education and health. In this paper, we provide a formal model and theoretical concepts for PA systems. We also propose a new location model called DPVW-model for annotated entities. DPVW-model represents aspects of the annotatable physical world as well as the virtual domain. Then, we discuss the implementation and usability study of our system. Finally, we compare it with other similar systems.

Keywords: Mixed reality, physical annotation, location model, Internet information system

1. Introduction

Mixed Reality (MR) provides the capability of merging the real world with the virtual world. It can be used to digitally enrich the real world around us, which will help to improve different aspects of modern life such as learning, gaming, shopping and more. MR already exists for many years, but recently has been used widely for mobiles especially with the portability and advanced capabilities of current mobiles phones (e.g., GPS, camera, connectivity, accelerometer and digital compass).

Based on different uses of MR, it may have different names such as Augmented Virtuality, Augmented Reality, geo-tagged information, physical labeling and physical annotations [18,20,24,26,28].

In our research, we focus on annotating the physical world which aims to add digital information to physical entities. Physical Annotation (or PA, for short) is one of the most important techniques in pervasive computing for providing useful information about annotated entities, whether it is a small object such as a cup, a person, or even a geographical spot in a city. In previous work [1], we provided a literature review of the different technologies and uses of PA.

We found there are many systems in this area, such as Yellow Arrow [30] which is one of the significant PA systems in the early last decade. It used a yellow marker on annotated entities in order to grab attention for those entities and also to add tag numbers to them. Also, recently there are Layar [15] Wikitude [27] and Junaio [14], which use the latest pervasive computing technologies such as GPS coordinates and image recognition techniques.

With the increasing demand and the popularity of using PAs, there are many challenges arising such as managing and structuring the annotated entities in the real world, and organizing their annotations. Also, there is a need for getting dynamic annotations within different context awareness situations.

This paper proposes an approach to PA systems which can be achieved through enhancing four aspects of PA systems and the services they provide:

* Corresponding author.

formalizing PA systems towards a foundation for this type of information system, modeling context-awareness aspects in PA systems, providing location models for outdoor and indoor annotated entities, and defining a range of possible relationships between annotations and targets. This paper provides a formal model for PA systems (which we call ALPHA), and also defines the different types of annotated entities (targets). Moreover, the paper discusses the notion of what we call Dynamic Physical and Virtual World models (or DPVW-models, for short) for physical world modeling of outdoor and indoor entities, and also describes the physical model and virtual model links. This model will help to manage and organize the annotations of physical entities in an efficient way, and give the system more features for creating and retrieving annotations. In particular, PAs are then seen as "decorating" a DPVW-model, which is an abstract simplified representation of the physical world. And the same physical space can have multiple DPVW-models associated with it. Finally, as a proof-of-concept of our approach of basing PA system design on our formal model and the DPVW-model, we implemented a PA system (which we call ALPHAsys) that can help users annotate physical entities efficiently. Then, we discuss a usability study of our system, and compare it with other systems.

This paper is organized as follows. Section 2 presents a summary of our formal PA model called ALPHA, and Physical Annotation concepts. Then, we provide and discuss our DPVW-model in Section 3. Then in Section 4, we discuss the relationships between annotations and annotated entities. Then, Section 5 provides a generic PA architecture and our prototype of ALPHAsys. In Section 6, we present the usability test of ALPHA and compare it with the popular Layar system. Section 7 presents the related work. And then the conclusion is in Section 8.

2. A conceptual model of physical annotations

We design our PA model, which we call Advanced Location-aware Physical Annotation (ALPHA) model as comprising three main parts: (A) the annotation part, which includes the content, format, type and so on (see below); (B) the physical entity (or the target) being annotated such as a location, or a small object; and (C) the link between the annotation and the annotated target, which includes the conditions of the relationship, anchor properties, and so on.

A. The annotation

The annotation part comprises the following properties: the annotation ID, annotation type, the annotation content itself, users, groups, and author category, as explained below.

A.1. Annotation ID

This ID is a unique identifier for each annotation.

Definition A.1 (Annotation identifier)**.** We define AID as a set of annotation identifiers, and aID ∈ AID denotes an annotation identifier.

A.2. Annotation type

This property concerns the purpose of the annotation. Therefore, based on those purposes, there is a need to classify these annotations. Examples of annotation purposes/types are "educational", "commercial", "health", "security" and more, and could be based on a domain-specific ontology.

Definition A.2 (Annotation type)**.** We define AT as a set of annotation types, and at ∈ AT denotes an annotation type. The set AT = {education, tourism, health, security, commercial}. (Other types can be added but this is an initial proposal based on observing PA application areas).

A.3. Annotation content

This property contains the actual information and the real content of the annotation.

Definition A.3 (Annotation content)**.** AC is a set of possible annotation contents (e.g., as text or in a format as defined by an XML schema, or a set of video clips) and ac ∈ AC denotes a piece of annotation content.

A.4. Users

This property includes all the system users, including authors and readers.

Definition A.4 (Users)**.** Let USER be a set of users, and user ∈ USER denotes a generic user.

A.5. Group

The aim of this property is to enable organization of users and classify them into groups based on their interests. Moreover, the author will have the choice to address his/her annotation to a particular group. For example, there could be educational annotations which were addressed to high school students, so that

the only users who can access these annotations are such students.

Definition A.5 (Groups)**.** GR ∈ 2^{USER} is a set of groups of users. And gr ∈ GR denotes one group of users.

A.6. Author

This property describes the identity of the author who can create, delete or modify the annotation.

Definition A.6 (Authors)**.** We define AU as a set of authors of annotations, and au ∈ AU denotes a generic author, and AU ⊆ USER.

Based on the above definitions, we can now define an annotation as follows.

Definition A (Annotation)**.** An annotation ann is a 6-tuple of the following form:

ann = <aID, at, ac, au, user, gr>,

where aID is an annotation identifier, i.e. aID ∈ AID, at is the annotation type, i.e. at ∈ AT, ac is the content type, i.e. ac ∈ AC, au is the annotation's author, i.e. au ∈ AU, user is the annotation's user, i.e. user ∈ USER, and gr is the user's group, i.e. gr ∈ GR. Any one of these values (except aID) can be null or defaulted to a fixed value.

B. The target

The target part describes the annotated entities (or targets) which exist physically in the real world. Instead of having one single target or annotated entity at one time, in our PA formal model, we have divided the targets into three different types of annotated entities, the first one is the single entity which always is one piece of physical object such as a cup, a person, or even a building. This type of annotated entity is the simplest type of entity, which also means they are easy to represent in a location model and easy to deal with in terms of creating, retrieving and querying annotations. The other type of the target is composed of more than one entity; this target type may be called a collection of entities, and is often more complex in terms of creating, retrieving, or querying their annotations. Also, a collection of entities often have similar properties in different ways which could be the colour, structure and so on. For example, the user may conceptually "make" a collection of cups in his/her room and give them a general annotation "my cups". So, a main feature of the collection is that it should be located within the same physical location (another example are books on the same bookshelf). What ultimately defines single and collection entities is how the person leaving the annotations for the entities view the entities.

Another target type is the virtual group which refers to a group of entities physically existing in the real world, but generally not physically grouped together in one location, and may be far in distance from each other. The entities in a virtual group are grouped together via relationships between entities, or via sharing context, such as a virtual group that comprises all the user's possessions in the whole building in his/her workplace, independent of where these objects are located in the building. This target type is associated with the idea of the annotation mapping property in our PA formal model, which means, for example, one annotation is linked to three different entities at three different locations in the location model.

A virtual group is also associated with the context dependency property in the PA formal model. Context dependency (also elaborated on later) means that one annotation is linked to a target entity (which can be single, collection or virtual group); but this linking depends on different contexts or circumstances. For example, the annotation will be visible only if the annotated entity is near-by another specific entity, or person. So, a main feature of the virtual group is that, unlike a collection, it shouldn't need to have entities in close physical proximity or existing within the same small physical area (with respect to an a priori chosen location model). Figure 1 shows examples of a collection, which is a desk, chair and books which are located in one room. The figure also shows a virtual group which comprises person 2, his/her cup and his/her room.

B.1. Target ID

This property describes the unique identifier of the annotated entity.

Definition B.1 (Target identifier)**.** We define t_id as a set of target identifiers, and t_id ∈ T_ID denotes a target identifier.

B.2. Target kind

This property aims to describe the nature of the annotated entity. The entity can be a location, a small object such as a cup, Defining and classifying the target's kind will provide better understanding of the possible annotations that could be associated with that entity.

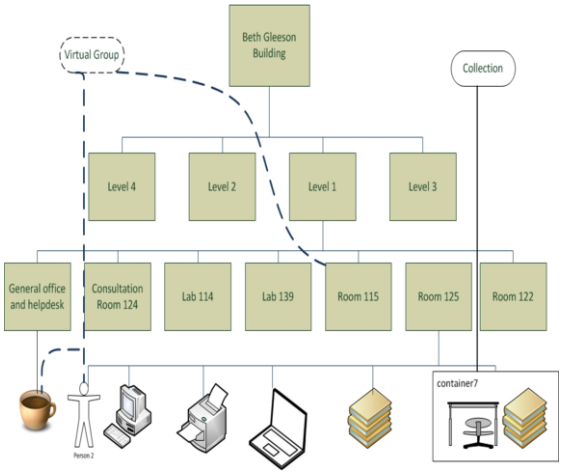


Fig. 1. Examples for collection and virtual group.

Definition B.2 (Target kind)**.** We define T_K as a set of target kinds, and t_k ∈ T_K is a target kind. The set T_K = {geographical_location, object, person}.

B.3. Target location

This property aims to describe the real physical location of an entity. We represent this property in our database by storing the GPS coordinates. Because some entities do not have a fixed location, this property is optional.

Definition B.3 (Target location)**.** We define T_LOC as a set of target locations, and t_loc ∈ T_LOC is a target location. The set T_LOC might be a set of geometric or symbolic coordinates (with respect to some location model).

Definition B (Target)**.** A target t is a 3-tuple of the form:

$$t = <t\_ID, t\_K, t\_LOC>,$$

where T is a set of all possible targets (as defined with respect to a model that we explain later) and t ∈ T is a target, T_ID is a set of all possible target IDs, and t_ID ∈ T_ID is a target identifier which can be a single value (or a set of identifiers), T_K is the set of all target kinds and t_K ∈ T_K denotes a target kind (or a set of target kinds as explained below), and T_LOC is a set of all possible target locations which includes the geometric or/and symbolic locations and t_LOC ∈ T_LOC denotes a target location (or a set of target locations as explained below).

As we have different types of annotated targets, the following are the corresponding definitions for the different types of targets, as explained earlier. These will help us later in defining the relationships between annotations and targets.

Single target: at = <t_id,t_k,t_loc>

Collection target: ct = <t_id's,t_k's,t_loc>

Virtual group target: vgt = <t_id's,t_k's,t_loc's>,

where t_id's is a set of IDs for the targets that belong to the collection or virtual group, t_id's is a set of target kinds for the targets respectively that are included in the collection or the virtual group, t_k's are the respective target kinds of the targets, and t_loc's is a set of target locations for the respective targets in the virtual group. Note that since a collection is located in one place, it will have only one location (with respect to a location model, which we explain later).

C. The link (a physical annotation)

We define a PA as the link or the bridge that connects the annotation component to the target component. As we discussed, the annotation part or the target part can be independent entities and can stand alone by themselves. However, the link is dependent on both the annotation and the target components. The following are the link's properties.

C.1. PA ID

This ID is a unique identifier for each link or PA. This ID is a unique identifier for each physical annotation.

Definition C.1 (Link/PA identifier)**.** We define paID as a set of physical annotation identifiers, and paID ∈ PAID denotes a physical annotation identifier.

C.2. Annotation

This property of a PA is the annotation part, whose structure is as described in Section A.

C.3. Target

This property of a PA is the target part, whose structure is as described in Section B.

C.4. Mapping

Mapping is a property that describes the relationship between the annotation part and the annotated entity(-ies) or target part.

Definition C.4 (Mapping)**.** We define MP to be a set of mapping types, namely, {one-to-one, one-to-many, many-to-one, many-to-many}, and mp ∈ MP denotes a mapping type.

C.5. Context dependency

This property describes all the possible conditions and situations that make the annotation readable and accessible to the users. The annotation could be static, which means it will be available every time and under any circumstances. However, the annotation could be dynamic, which will be available only under certain conditions and circumstances. Therefore, context dependency will contain all the conditions that allow the annotation to be active and available to the users. An example of a dynamic annotation is when an annotation depends on many contexts such as location, time and weather conditions. So, all the three conditions must be met in order to be visible or active to the users. The context dependency also includes the dependency between the entities for annotation access – the idea is that an annotation can only be read when all entities are detected.

The value in this property can vary from one annotation to another and may have many values, so the following set of values for this property is just one possibility.

Definition C.5 (Context dependency)**.** We define CX as a set of contexts, and cx ∈ CX is a context (or a type of context information). CX = {location, time, date, nearby person, nearby object, entity dependency}.

The following is a general definition of a physical annotation.

Definition C (Physical annotation/link)**.** A physical annotation (or link) is a 5-tuple of the following form:

$$l = \langle paID, ann, t, cxD, mp \rangle,$$

where paID is the unique number to identify each PA, i.e. paID ∈ PAID, ann describes all the properties which belong to the annotation itself (as in Definition A), i.e. ann ∈ ANN, t refers to the target, which is the annotated entity (as in Definition B), i.e. t ∈ T, cxD is the annotation context dependency, i.e. cxD, is a subset of CX, (in practice cxD are attributes with particular context values) and mp is the mapping property which presents the relationship between the annotation and the entity, i.e. mp ∈ MP.

3. Dynamic physical and virtual world model (DPVW-model)

The formal model shows that the PA system may have different types of targets based on its constructions. The target can be a single entity which is only one annotated entity, or a collection of targets, which are located in one place, or a virtual group, when targets are located in different places.

Therefore, it is necessary that the location model should present these different types of targets. As a result, we present the Dynamic Physical and Virtual World model (DPVW-model). We assert that there are three important key aspects in a world model (which includes a location model). The first one is the dynamic or temporal nature of entities. And the second one is the notion of physical collections, and the third one is the virtual containment or relatedness. In our conceptualization of a physical annotation system, the environment can be dynamic which means the annotated entities can be moved from one place to another, or in other words, the physical locations of some entities are temporal. Some entities may change their location frequently, such as a person. Other entities may change their location rarely. For example, a chair in a room can be moved to another location in the environment, whereas other entities cannot be moved to another place such as a building. Therefore, with respect to a PA system, it is useful to define all the possible annotatable entities for the system, i.e., we can build a location/world model that includes such entities for a PA system. Relationships among annotations can also be indirectly determined by, or are induced by the relationships among, the targets they annotate. Also, a DPVW-model can be used as a basis for browsing and querying annotations. As we discuss in previous work [2], there are many different entities that can be annotated in a PA system. Moreover, the relationships between the annotations and the annotated entities are different and there are many possible scenarios. Also, the context dependency can include different elements such as time, near-by entities, and so on. However, we state and illustrate the main types of annotatable entities that can be considered when one builds a world/location model, as follows: (i) outdoor locations and their structure: in forms such as point, line, and polygon, (ii) indoor locations and their structure: e.g., room, floor, corridor, (iii) objects indoors and/or outdoors, some movable, some not: e.g., chair, wall, person, moveable object, (iv) collection of locations or objects, and (v) virtual groups of locations or objects, as described earlier.

Outdoor location A point means a geometric location which can be described using coordinates. In outdoor points, we would often use the GPS longitude and latitude to describe this point. A line con-

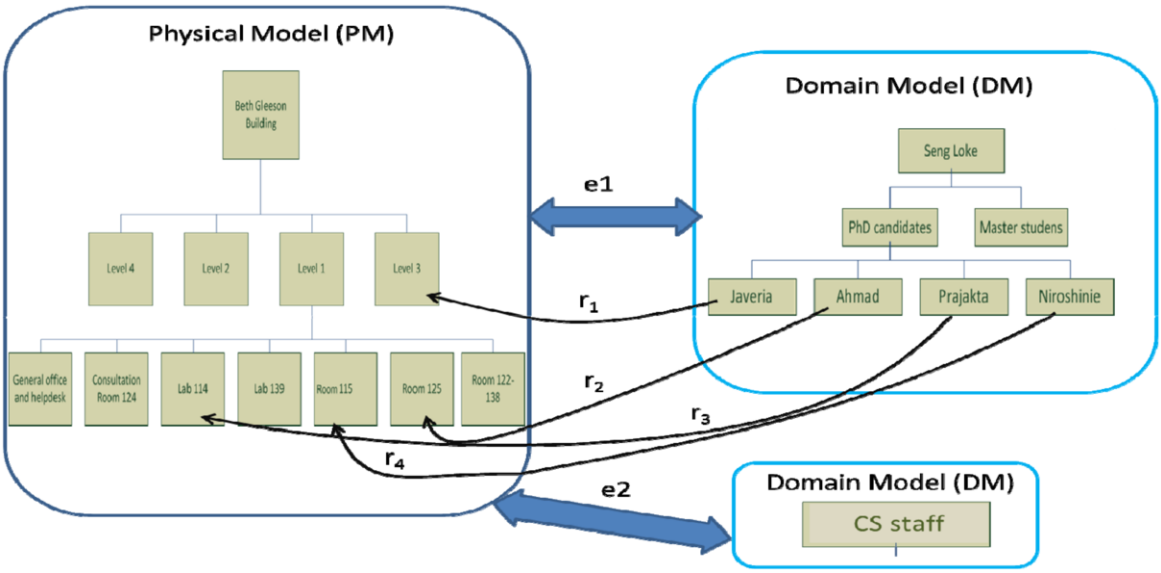


Fig. 2. DPVW-model for BG building and Seng's students.

nects two points, so the system stores the coordinates of the two points/ends of the line. An example of this is a street viewed as a line. A polygon is a series of connected lines and often comes as an irregular shape. An example of this is a block in the CBD, or a building. One can annotate points, lines and polygons once defined. A user can define a point to annotate by taking its GPS coordinates (e.g., the current location) and then leaving an annotation to be associated with that point/location. A user can define a line by picking two points (or their GPS coordinates) (either via a map application or walking from one point to the other) and then associating an annotation with the line, or similarly, define a polygon by specifying a set of lines or points (e.g., walking to them and taking the "current location" readings, respectively) and then leaving an annotation for the polygon. Similarly, another user can read such annotations while at that point, someone along the line or within the polygon.

Indoor location An indoor location is often a covered area and that includes the interior of a building. An example of an indoor location is a room, or a whole floor, which can be described with respect to an indoor location model (which might depend on the indoor positioning technology used). Indoor locations can have structure. Figure 1 illustrates the structure of the Beth Gleeson building, and so, the different locations within this structure are annotatable.

Outdoor and indoor objects/persons Objects or persons must be identifiable to the PA system, e.g. via RFID tagging, 2D barcodes, or image recognition via a smartphone camera. A movable entity, in general, has two dynamic features, size and the position. Examples of a movable entity are a table, or a person. A user can scan a 2D barcoded (or an RFID tagged) object and leave/read an annotation for it.

Collection This could be a composed target which can be more than one entities located in one physical location; as we discussed the target in previous section, this entities tend to have a similar property such as their owners.

Virtual Group (VG) Entities could be grouped virtually, even if there are located in different physical locations, and similar to collections, these entities might have similar features, such as type of product, or colour. VG can be simple and independent as shown in Fig. 1, however VG can be also part of another virtual group or domain model (DM), as shown in Fig. 2, PhD candidates with their supervisor Dr. Seng make up a virtual group as shown in the Virtual/Domain Model (VM) (as called in the figure, the Domain Model (DM)). Also, candidates are located in different places as shown in the Physical Model (PM) of the Beth Gleeson Building. Therefore, the DPVW-model can present annotatable entities in a PM or/and a VM, and link the VM to the PM.

Therefore, the DPVW-model contains two parts; first one is Physical Model (PM), which represents the real physical world, and the second one is Virtual/Domain Model (DM), which represents the virtual group, or logical structure of entities, and/or even the virtual entities such as the "master students" node in Fig. 2. In the DPVW-model, as shown in

Fig. 2, we have on the left side a PM of the BG building in La Trobe university, and the right side shows the DM/VM of Seng Loke's students. The relationships between the PMs and DM/VMs are depicted with the thick arrows (e.g., labeled "e1" and "e2"), and the thin arrows represent the relationships between particular nodes in a DM/VM to particular nodes in a PM. For example, r1 links the student "Javeria" on the DM to her location on level 3 of BG building. Also, r2 links "Ahmad" to his location in room 125 on the PM of the BG building. The same with r3 which links "Prajakta" to her location in lab 114. Finally, r4 links "Niroshinie" to room 115.

Moreover, for the same PM, we can link another related DM such as a domain model that represents the maintenance staff or Computer Science lecturers and so on. Conversely, DM also can be linked to multiple PMs.

The idea is that a person can leave/retrieve annotations that are directly associated with physical entities in a PM, as identified by the entities' location (e.g., GPS coordinates), and/or tags (e.g., 2D barcode or NFC tag), or a person can leave/retrieve annotations that are indirectly associated with physical entities, via the nodes in a VM/DM. For example, Seng's PhD students can be annotated with one annotation by annotating the PhD candidates' node in the DM of Seng, so instead of annotating each node individually, users can use VM/DM models to annotate a collection or a VG in just annotating their parent (more details will be explain in the implementation section).

Figure 4 shows a DPVW-model of an outdoor shopping mall, namely, the Bourke Street Mall, in Melbourne, Australia. We divided the mall into four parts: Northeast (NE), Southeast (SE), Southwest (SW) and Northwest (NW)) as shown in Fig. 3. Then, we created the DPVW-model for the mall as shown in Fig. 4. Of course, there could be other ways of dividing up Bourke Street Mall yielding different DPVW-models over the same geographical area – but we use Fig. 3 as one possible illustration. The DPVW-model not only serves as a reference for what is annotatable at design-time but also at run-time, as browsable models for users to retrieve annotations, so users can browse annotations for a current node, or for its higher level, or even the kids nodes of it. It also easily and efficiently supports annotating collections VGs. Different DPVW models are used depending on the application. The ALPHAsys system could have a collection of DPVW-models for different annotatable entities and spaces.

Fig. 3. Bourke St. Mall divided into four top-level zones.

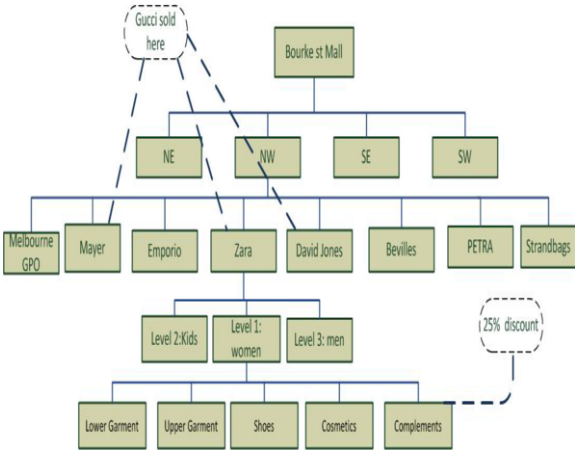


Fig. 4. A snapshot of the DPVW-model for the Bourke Street Mall, with some example annotations.

4. Types of physical annotations: Relationships between annotations and targets

There are different possible relationships in a PA system between annotations and targets, and different conditions (if any) of linking them together, thereby inducing a typing on physical annotations – we describe seven examples of PA/link types below, though they are not exhaustive. These PA types are specializations of Definition C.

Type 1: Basic annotation.

Definition. This type is a PA of the form:

<paID, ann, t, null, one-to-one>.

This means there is only one annotation for one annotated target. This link here is in a static mode, which means the validity of this annotation for that target is all times and everywhere, without any conditions.

Example. A user annotates a chair with annotation "belongs to Ali", so the user can retrieve that annotation anytime and anywhere.

Type 2: Basic annotation with context dependency.

Definition. This type is a PA of the form:

<paID, ann, t, cxD, one-to-one>.

This is similar to the first type of the relationship when there is only one annotation for one annotated target. However, the link here isn't static; we call it a dynamic link, which means the validity of this annotation for that target is only under some conditions. In this type, the condition is one of a context dependency, so the annotation is visible or readable by the user only if conditions have been met (details of conditions are assumed specified in the operation of the PA system).

Example. Ali himself was annotated with annotation saying he works as a "technical support for CS Dept". This annotation will not make sense if users can read it every time and everywhere when they see Ali. So, the validity of this annotation is restricted by time and location. Validity time is only between 8am–5pm Monday to Friday, during typical work hours. And the validity location is only the Department of Computer Science in La Trobe University, for instance.

Type 3: Collection annotation.

Definition. This type is a PA of the form:

<paID, ann, ct, null, one-to-one>,

where ct refers to a collection target (as defined in Section 2).

Here, there is one annotation which is linked to many targets grouped together as one collection; hence, the mapping type is still one-to-one. In other words each entity of the collection individually do not make that annotation valid, i.e., the annotation is not for any one of them alone, so they have to be physically together to make sense and to make that annotation sensible. This link here is static, which means the user can access the annotation without any conditions.

Example. Assume the user annotates the "whole campus of Latrobe University". All buildings, sport facilities, shops and car parks make up the collection. The annotation is then not valid for one part alone, but for the whole university, it is valid – one needs to scan all parts of the university (or be within the campus) before the annotation can be viewed; scanning just one of the buildings will not bring up the annotation.

Type 4: 1-M Mapping annotation.

Definition. This type is a PA of the form:

<paID, ann, vgt, null, one-to-many>.

This type of link or physical annotation means there is one annotation linked to many targets, so that the annotation is a true statement and makes sense for each target individually. And that happens when these targets have a similar property, in order to be linked to one annotation. This link here is static, which means the user can access the annotation without any conditions. Unlike collection annotations above, users can get the (same) annotation when the targets are far from each other, and need only scan any one of the annotated targets in the virtual group to get the annotation.

Example. With "belongs to Ali", Ali annotates all his belongings such as his car, books, rooms, chair and pens. So, this annotation is linked to many entities, and can be retrieved by scanning any one of them individually.

Type 5: Collection annotation with context dependency.

Definition. This type is a PA of the form:

<paID, ann, ct, cxD, one-to-one>.

This type is similar to the third type which is one annotation linked to a collection of targets that are physically grouped and exist together; however, the difference here is that the link is not static, which means there is at least one condition allowing access to this annotation. So, to retrieve the annotation, users must meet the context conditions.

Example. Assume we have a Horse Drawn Carriage, so both the horse and the carriage make one collection. The context dependency (condition of linking) contains three elements: a time, a location and the weather. The validity time is only from 9pm till 10pm, location availability is only the Melbourne CBD and the weather must not be raining. An annotation says "you can hire me and take a tour around CBD for $100 up to 6 persons", but the annotation is valid only when (i) the component targets (horse and carriage) are physically together (given the concept of the collection), and (ii) when the context dependency conditions have been met.

Type 6: 1-M Mapping annotation with context dependency.

Definition. This type is a PA of the form:

<paID, ann, vgt, cxD, one-to-many>.

Similar to the fourth type where the relationship is from one annotation to many targets in a virtual group, but the target here does not make one collection (as we define "collection" above) and shouldn't need to be physically located in one place. Also, here, the annotation is not static and it has a context dependency and conditions to be valid.

Example. In the La Trobe university library, there are some employees who are dedicated to helping students to access books on the shelves. So the annotation on such staff is "you can ask me if u need help to get a book". The target here is more than one employee. The context dependency is location, so that the annotation on the staff is only active/visible when the employee is near-by the bookshelves, and also time, when the employee is on duty. Another example is when users annotate targets useful for blind people; these targets don't have to be at the same physical location, and the context dependency here is the blind person him/her-self (only the blind accesses such annotations, which are read to them).

Type 7: Multi-annotation (M-1 mapping annotation) with common context dependency.

Definition. This type is a PA of the form:

<paID1, ann1, t, cxD, many-to-one>,
where there must be at least one other PA of the form:

<paID2, ann2, t, cxD, many-to-one>.

This type of relationship between the annotation and the annotated entity is when one target (which may be a single, collection or virtual group) is linked to from more than one annotation. And the annotations could be static annotations which mean they are valid without context dependencies (cxD is null) or there could be dynamic annotations, which mean they are only available under some context conditions (and the same context conditions are used for all of the annotations). Of course, one can define another type of PA where context conditions need not be the same for each annotation.

Table 1
Relationships between annotations (A) and targets (T)

A	T	Type of physical annotations	**S**ingle, **C**ollection, **V**irtual **G**roup	In Fig. 5
1	1	Basic annotation	S	Ann 1
1	1	Basic annotation with context dependency	S	Ann 2
1	M	Collection annotation	C	Ann 3
1	M	Mapping annotation	VG	Ann4
1	M	Collection annotation with context dependency	C	Ann5
1	M	Mapping annotation with context dependency	VG	Ann6
M	1	Multi-annotation with common context dependency	S, C, VG	Container7

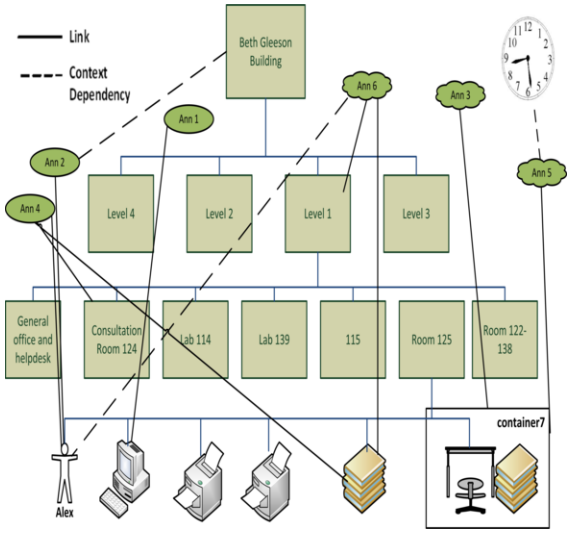


Fig. 5. An example of a DPVW-model with possible relationships between targets and annotations.

Example. Assume we have a cup which is a single target. We can annotate it by many annotations such as "red colour", "belongs to Alex", "this is used to drink tea only", and so on.

Table 1 and Fig. 5 illustrate all of the previous seven types of the relationships between the annotations and the targets in our location model. In Fig. 5, ann1 represents the first type of PA, the basic annotation, when one annotation is linked to the PC without any constraints. Also, ann2 represents the second type of the relationship which is, basic annotation with context dependency, so that the annotation for "Alex" in the figure can be retrieved as long as the retriever is inside the BG building. The next annota-

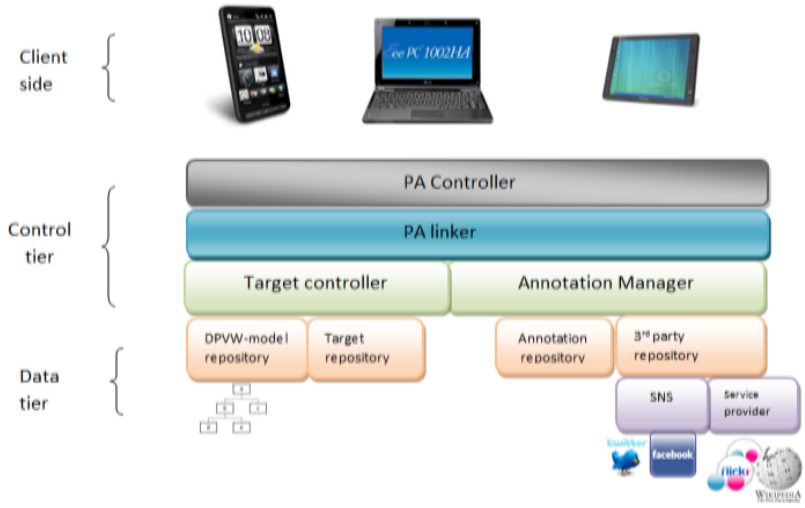


Fig. 6. High level generic architecture for PA systems.

tion is ann3, which represents annotating a collection which is called container7. It contains books, a chair and a desk. The annotation ann4 shows the mapping annotation type, where one valid annotation is linked to books, "Alex" and "consultation room 124"; such an annotation is "today seminar will be held in consultation room 124 by "Alex" about these books…". The next annotation is ann5 which shows annotating a collection with context dependency, so that users can retrieve ann5 only at particular times. The annotation ann6 illustrates the type of mapping annotation with context dependency; ann6 is valid for both the books and level 1, so the annotation can be retrieved only if "Alex" is nearby any one of them. The last relationship type can be seen for the collection "container7", so that more than one annotation is linked to it, which are ann3 and ann5.

5. ALPHAsys system design and implementation

ALPHAsys is an application for physical annotations based on the ALPHA model, which allows digitally annotating the real physical entities, whether they are small objects, persons or locations, with real data. The entities could be tagged with three options individually or combined with others, i.e. by using 2D QR codes, image recognition, and/or GPS coordinates.

ALPHAsys is an implementation of the generic PA architecture shown in Fig. 6, reflecting the ALPHA model. The architecture can be viewed based on the formal model as comprising three parts; the first part is the annotation part starting from the Annotation Manager down to the SNS and the Service Providers, as follows:

1. *Annotation manager*: this layer is about the annotation content, stored in two repositories:

Annotation Repository: which contains the system's own annotations written by users, and External 3rd party annotation repositories: the annotation content can be provided by third party providers, such as social network services. For example, Facebook can provide some useful annotations as well as Wikipedia; so, this layer refers to such information services.

2. *Target controller*: it includes:

(i) Target Repository: that contains information about the annotated entity, whether it is a small object, location or person. It also stores the state of a target whether it is a single, collection, or a virtual group. The repository includes also the entity's target kind which will help to determine the possible associated annotations of this object.

(ii) DPVW-model repository: stores the DPVW-model (which includes the physical location model and the virtual model of spatial targets as we discussed in Section 3) and tracks the targets within the DPVW-model, e.g., their locations in the location model (e.g., within a room on a floor in a building), giving the user the option to browse the DPVW-model to retrieve the parent node's annotations as well, or the kids nodes' annotations. For example, if the user is in room 219 on the 2nd floor in

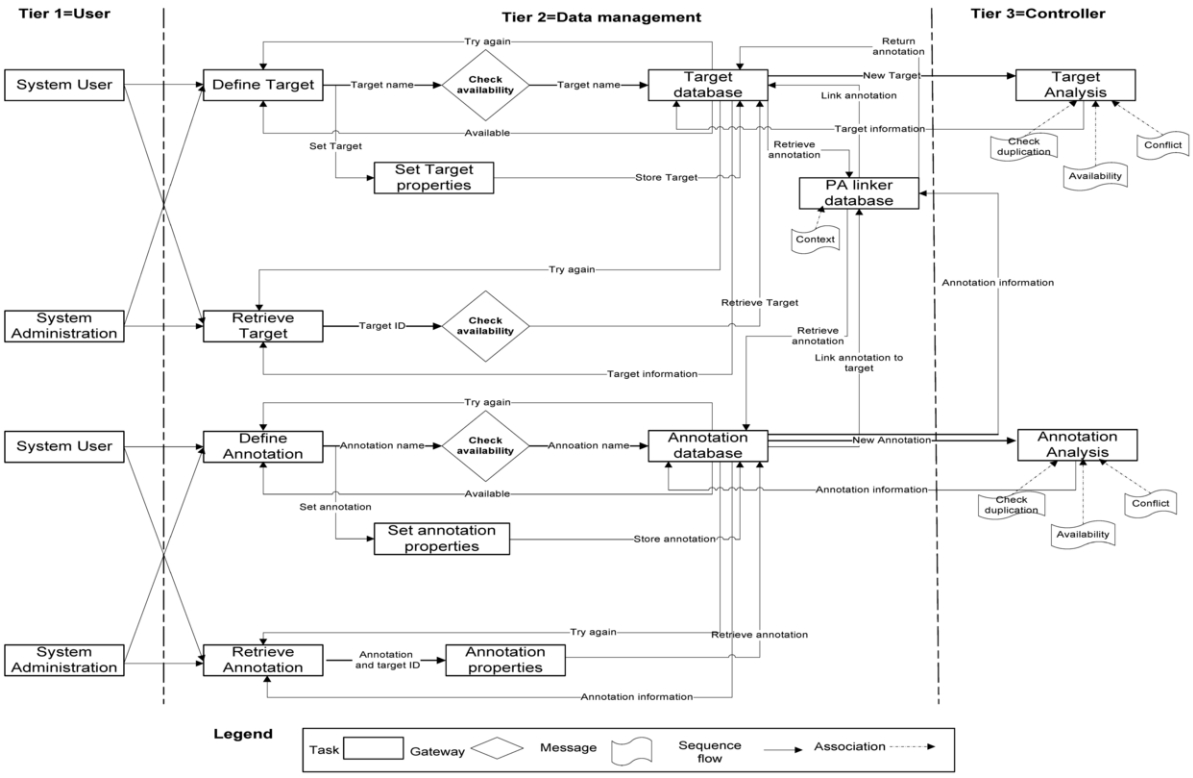


Fig. 7. ALPHAsys architecture.

building B, we can present this location model as a spatial tree to the user. The user may then like to retrieve the annotations for that room and the floor as well. So, this will locate the entity in the location model hierarchy and give the user the option to retrieve the annotations belonging to the space which contains the location, not just the annotations for the spot/location.

3. *Link part*: this includes:

(i) PA Linker: the main job of this layer is to map annotations to entities/targets. This tier includes conditions of the linking, and uses the PA link properties such as context dependencies and mapping. It manages annotations linked to collections and virtual groups.

(ii) PA Controller: this component controls the PA access; after the system retrieves the annotation from the lower tiers, it manages the interference/collision (if any) [4], and all possible situations that may affect the annotation.We also can view the architecture based on the server and client sides. In our design, we have chosen to reduce the complexity on the client side for many reasons including the possibility of limited resources on the client. Therefore, most operations are done on the server side. On the client side is the interface to the system on the user's handset, and by using a mobile phone, a laptop or a tablet, users can access, retrieve and modify physical annotations. On the server side, the server here includes the main components in Fig. 6.

On the client side, users can perform the following main operations and queries on the system: (1) annotate an entity (e.g., object, place, person, collection, virtual group), (2) grab and view annotations (e.g., browse location models, and retrieve annotations for single, collections, and virtual groups), (3) manage targets (i.e., add and modify a location model, add collections and virtual groups), (4) manage annotations (i.e., update and delete annotations), and (5) change settings (i.e., add users, add groups, and add users to groups).

As shown in the architecture in Fig. 7, what the user has to do to annotate an entity or retrieve annotations is as follows:

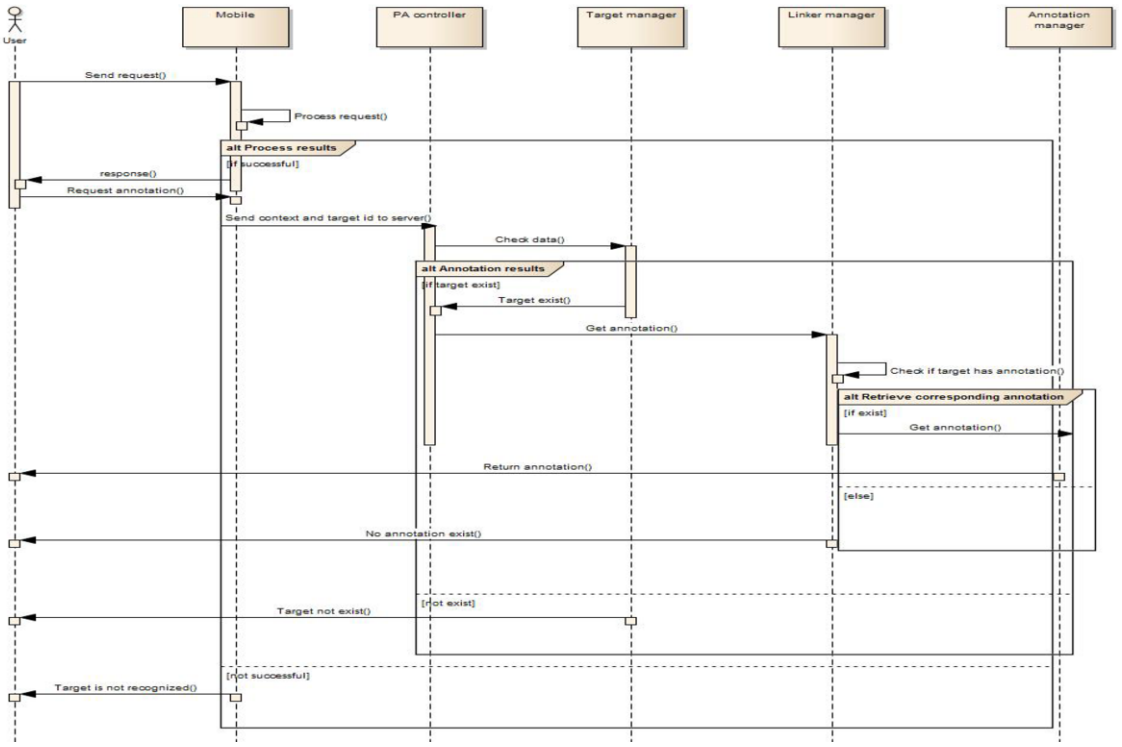


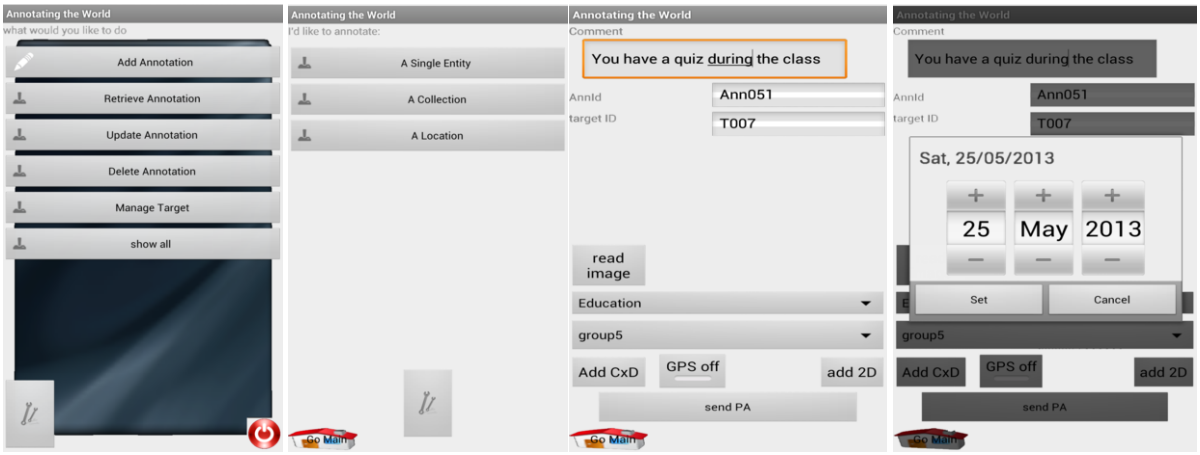
Fig. 8. PA retrieval in ALPHAsys.

(1) The target must be added into the DB, users can add one target, collections of targets, or virtual groups. That can be done by scanning a 2D barcode on the target, image recognition by pointing the mobile device camera at the object and matching (via the Vuforia SDK[1]), using the target's GPS coordinates (if the target to be annotated is a location), or a combination of some of them. Also, we can use Bluetooth or RFID to associate annotations with targets, although we haven't implemented Bluetooth or RFID tagging here.
(2) Then the user can annotate the target by specifying whether access to that annotation is "simple" or "complex" (complex is where the user then needs to specify context dependencies).
(3) To retrieve an annotation, users should meet the conditions of annotation access; for collections of objects, for example, the user needs to scan all the objects in the collection (and fulfill any other context conditions) before retrieving the annotation for that collection.
(4) Users have the choice to retrieve annotations for one target, collections, or browse the location model to obtain annotations.

Figure 8 shows the sequence diagram of reading a target and retrieving annotation(s) for it. The process starts by sending a request by a mobile user to the mobile in order to read and recognize the target, this process can be done by reading the QR code, GPS coordinate, or image recognition of the target. Then if the response was successful, the mobile sends the request to the server side, which is received by the PA controller which, in turn, will check with the target manager if the scanned target exists in the target repository; if so, the PA controller will check with the annotation manager if there is an annotation linked to that target, and if so, the PA controller will retrieve the target's annotations from the annotation manager based on the user's current context.

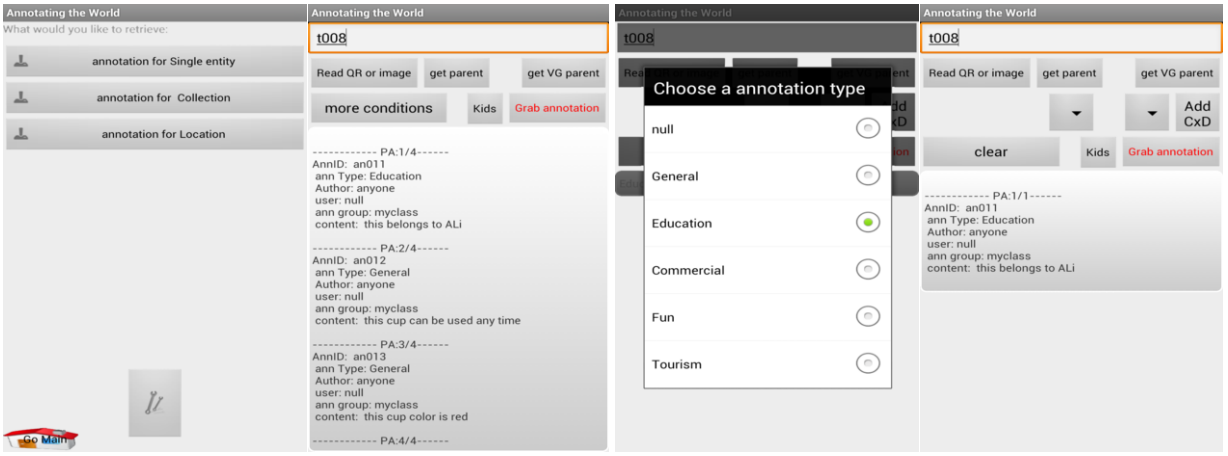
Figure 9i shows the main screen of ALPHAsys. Figure 9ii shows the options of creating annotations, whether a single entity, a collection of entities or a geographical location. Figure 9iii shows the interface for leaving an annotation for a single entity where users can choose the group, annotation type, adding

[1] https://developer.qualcomm.com



(i) (ii) (iii) (iv)

Fig. 9. Screenshots illustrating leaving annotations using ALPHAsys.



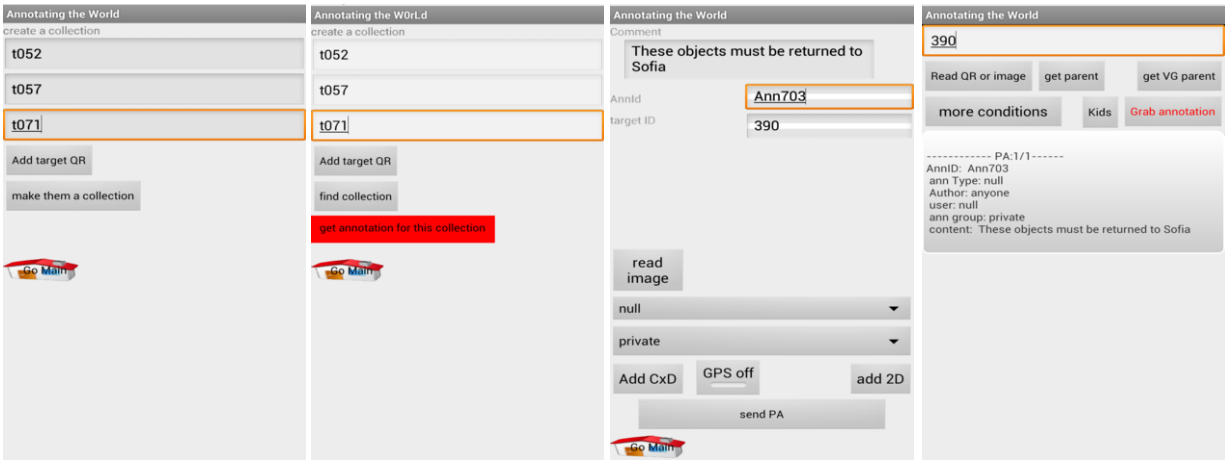
(i) (ii) (iii) (iv)

Fig. 10. Screenshots for retrieving annotations through image recognition or 2D barcode.

GPS coordinates and context dependencies (e.g., a date as shown in Fig. 9iv).

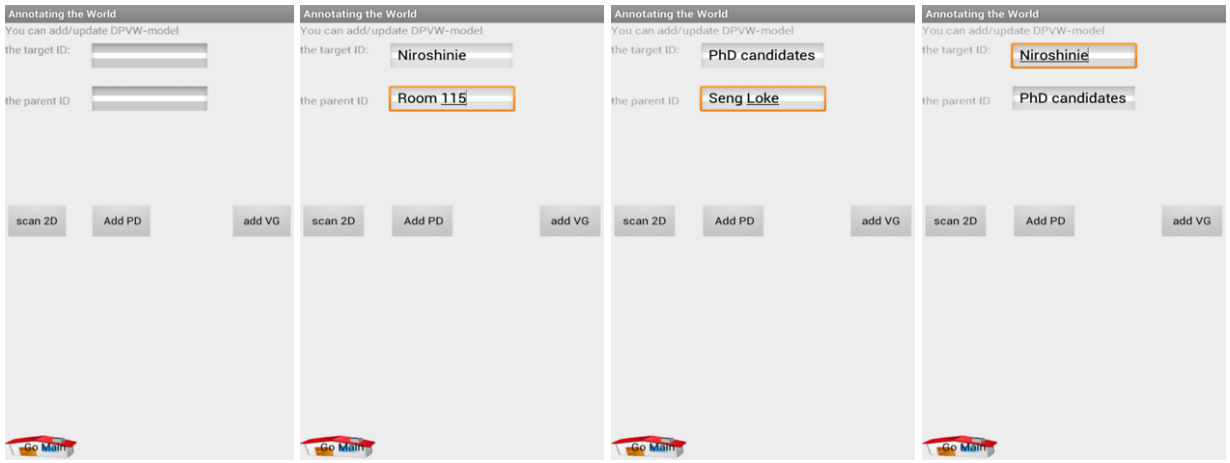
Figure 10 illustrates the retrieval process. Figure 10i shows the retrieval options so users can choose retrieving annotation for a single, a collection, or a location entity. Then, Fig. 10ii shows the user retrieving annotations for the target "t008" without specifying the annotation type. The result will be bigger than the result in Fig. 10iv where the user specifies context dependencies to retrieve (or effectively, filter) annotations for a target, or annotation type as shown in Fig. 10iii, which in the figure leads to retrieving only the annotations of the required annotation type.

Also, note the button "get parent's annotation for this" is a means of browsing for annotations via the hierarchical location model (i.e., a DPVW-model as illustrated in Figs 1 and 4). Figure 11 shows how to make and retrieve annotations for a collection of targets. Figure 11i shows how to add targets to a collection by scanning the 2D code for each target; then, the user can click on the "make them a collection" button, which will assign a collection number to those targets and then the system goes to Fig. 11ii, where the user can add a general annotation to this collection – for example, as shown in Fig. 11iii, the annotation is "these objects must be returned to Sofia". Figure 11iv shows how to retrieve annotations for a collection: the user can scan each object one by one and click on the find collection button in order to check what collection the scanned objects make, and if they make a collection, the system will go to the



(i) (ii) (iii) (iv)

Fig. 11. Dealing with collections.



(i) (ii) (iii) (iv)

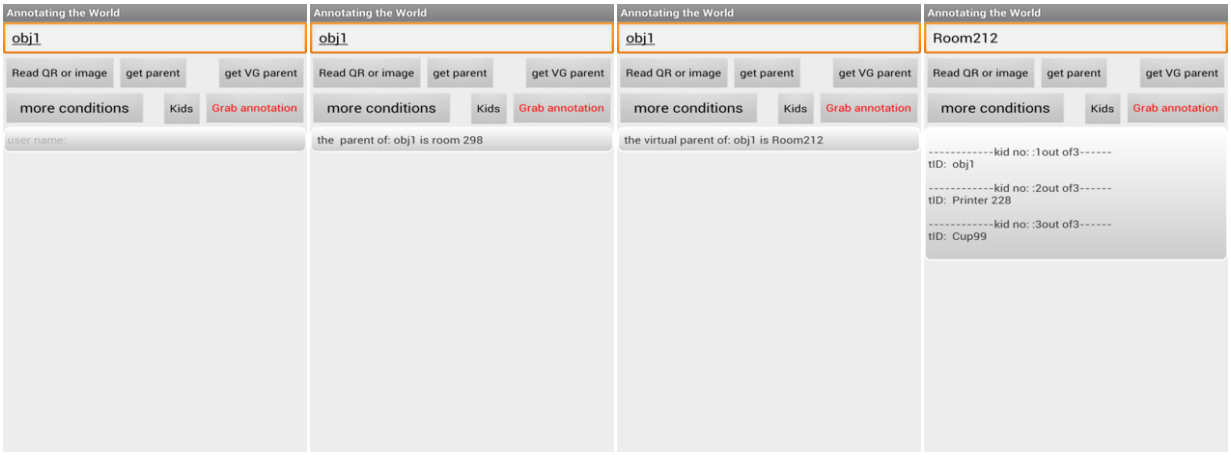
Fig. 12. Building the DPVW-model (virtual domain and physical domain) on the mobile.

retrieval screen with the collection ID. The other option of retrieving the collection is by entering the collection ID manually into the retrieval screen, but in this case the user should know the collection ID.

Figure 12i shows how a user (with appropriate permissions) can build new location models (that is, extending the range of annotatable targets by extending a DPVW-model or creating a new DPVW-model; though such models can be built beforehand by the developer or system administrator and loaded on to the mobile). Users can add a physical model (PM) or a virtual/domain model (DM), and can link a target to a physical parent or to a virtual parent (using the "parent" relationships in the DPVW-model). As we discussed in Section 3, Fig. 12ii shows how to add "Niroshinie" to "Room 115" which is located in the physical model PM of Beth Gleeson building.

Figure 12iii shows how to create a DM link for Seng by adding "Seng Loke" as a parent and "PhD candidates" as a child node. Then, Fig. 12iv shows how to add "Niroshinie" to a virtual group which is called PhD candidates, that is a part of Seng's domain model DM, as illustrated in Fig. 2. To add or update a PM or DM, users can add each node with its parent by using image recognition, 2D barcode scanning or using the GPS coordinates of each target, or users can enter the target IDs manually, as shown.

Figure 13i shows the main screen of browsing for annotations using the DPVW-model and showing some operations allowed with it. Figure 13ii shows



(i) (ii) (iii) (iv)

Fig. 13. Browsing annotations using a DPVW-model.

that the user retrieved the physical parent of "obj1" which is room 298 which will allow him/ her to retrieve the parent location's annotations. Figure 13iii shows that the user can also retrieve the virtual parent of that "obj1" which in this case is room 212. The same can be done with the example of Dr. Seng's students that we discussed before, so that when the user retrieves the physical parent for one of the students, the system will return the room number that he/she is in (i.e., the physical parent, or parent according the PM), but if the user retrieves the virtual parent of the student, the system will return Dr. Seng (i.e., the parent according to the VM/DM). Once the physical or virtual parent is retrieved, one can then view the annotation of the parent. Figure 13iv shows how the user can retrieve all the physical children or the virtual children nodes that come under an entity (in this case, children of room 212).

Figure 14 shows examples of targets that can be scanned by the ALPHAsys system using the mobile camera which can then be automatically retrieved and super-imposed with associated user entered annotations. For example, the user can scan the 2D barcode on the cup or scan the flowers via image recognition, and then see or type in an annotation for that object. For the image recognition, as was mentioned before, we used Vuforia AR SDK [25] which is a framework developed by Qualcomm. It allows mobile phones' cameras to recognize objects in live camera video streams based on images that have been stored on the cloud or the developer's local database. We preferred Vuforia AR SDK over other systems because it is more reliable and fast.

Fig. 14. Accessing targets through image recognition or 2D barcode.

Implementation and prototype As mentioned, we have created ALPHAsys as a prototype of a PA system based on the APLHA formal model and built a location model in order to test the applicability and feasibility of our model, and as a proof-of-concept. The implementation of the client has been done on an Android mobile "Samsung Galaxy S3", and the server side has been implemented as a RESTful API (HTTP REST interface to MySQL written in PHP) to communicate with the MySQL database on the server side. The common methods of RESTful are POST, GET, PUT and DELETE which are equivalent to Create, Read, Update and Delete (CRUD). As an example of using the API, users can get a target by its ID using the http GET method through this URL: http://homepage.cs.latrobe.edu.au/aaalzahrani/rest/Target/<tID>.

Table 2
ALPHAsys RESTful API

Resource	POST	GET	PUT	DELETE	Parameters
/Annotation	Create a new annotation	Retrieve a particular annotation	Update an annotation's properties	Delete an annotation	annID, annT, annUser, annAuthor, annGroup, content
/Target	Create a new target	Retrieve a target	Update a target	Delete a target	tID, tT, lat, lon
/DPVW-model	Create a new DPVW-model	Retrieve a target model	Update a target position in a model	Delete a target from a model	tID, parentID
/Link	Create a new Physical Annotation (PA) by linking annotations to the target	Retrieve a PA	Update a particular PA	Delete a PA	annID, tID, CxD

Users can also define new targets or update an existing target by posting parameters of targets, which are target ID (tID), target type (tT), and GPS obtained latitude (lat) and the longitude (lon) of targets. So, after this step, users can annotate new targets by sending annotation parameters to the server. Also, to define a new annotation, or update an existing one, users can do that by posting, to the server, annotation's parameters which are annID, annT, annUser, annAuthor, annGroup, content, and to retrieve or delete an annotation, users only need to post annID and context. Similarly when users want to link annotations to targets, they need to send the server annID, tID and CxD which is the context when users can retrieve the annotation for that target. Finally, to add, update, retrieve or delete targets to DPVW-model, they need to post tID and parentID.

Table 2 provides a summary of our REST API's main resources and parameters for each resource. The parameters reflect the definition of the ALPHA model.

6. Case study

ALPHAsys was deployed on two university campuses: King Abdualaziz University (KAU) in Saudi Arabia and La Trobe University (LTU) in Australia. Then, we obtained feedback on the usage after two weeks from users at both universities. The main aim of this study is to test the functionality and the usability of ALPHAsys in real environments. We want to see if ALPHAsys is useful and usable in the real world and to obtain general feedback on our prototype. It must be noted that we had not "beautified" the user interface but sought to test the functionality of the system with users. Also, the reason for choosing two different places was to test the impact of users' background on the ALPHAsys.

The test includes four stages: the first one is to understand participants' background for using pervasive technologies. The second stage is to try and let participants experience ALPHAsys. The third stage is to have participants answer a small survey based on their experiences. In the last stage, we compare the system with another PA system which is Layar in our case. The number of participants was 10 students, 5 in KAU and the other half in LTU. All of them are students from Bachelor degree to PhD degree, and from different majors, and so, they were given a short seminar to provide them background and explanation on ALPHAsys. The usage scenarios aim to allow them to create annotations with different contexts, and then to share them with others, and also to allow them to retrieve annotations in different contexts.

For anchoring, as mentioned earlier, we used three different techniques including QR codes, image recognition and GPS coordinates (RFIDs, Bluetooth and other anchoring systems could be used in the system, but we limited the system to the three techniques only). A number of 2D barcodes were placed on different entities, including small objects such as chairs, tables, PCs, bags and also on large entities such as rooms, buildings' entrances. The image recognition technique was used on some posters such as posters of final year student's projects.

The GPS coordinates were used only on outdoor locations. So, in order to link annotations to targets, users have the option to link it by using QR code, image recognition, or GPS coordinates. Users also can combine some of the linking options together For example, outside the entrance of CS in KAU, the main annotation that welcomes new students was associated with GPS coordinates only, so when users retrieve annotations based on the GPS coordinate, they will get the "Welcome to CS" annotation. There are also some QR codes associated with the same area (placed on the outdoor of the entrance and on

seats next to the entrance), so users can create annotations for each of the door or the seats by associating them with the same GPS coordinates, but with different QR codes. Therefore, in order to retrieve annotations for the seats, users have to scan the QR code and send it alongside with the GPS coordinates to the PA server (We assumed here the same seat's QR code is used for another seat somewhere else so that users have to send both GPS and QR code).

Participants installed the ALPHAsys Android app on their smartphones, and were given different situations and limited time to explore and understand the system. They also were allocated to different users' categories such as "Sameer's students", "group 5", "class 224", and "general", some of which were allocated to several categories. This helps to specify annotations for special groups and not for other groups.

Participants were asked to leave annotations on some entities, such as small objects and geographical areas, and associating them with different contexts and types of annotations such as "education" or "fun".

They were given six main types of tasks to do in order to test ALPHAsys as follows.

The first task is to create and retrieve annotations for different targets in indoor places whether small object or an indoor place without any context or constraints.

The second task is to create annotations and retrieve them, for the same entities in the first task, but this time with different contexts such as annotation type, time and location of retrieval.

The third task aims to create annotations and retrieve them for outdoor objects and locations with and without context.

The fourth task is to create a collection of entities and annotate them, and retrieve those annotations.

The fifth task aims to test the operations on the DPVW-model by adding targets or moving them to different locations in the physical model, and also to annotate different levels of the model. Updating or adding targets to the DPVW-model are supposed to be done by system administrators or targets' owners; however, for test purpose, we allow participants to add and update targets based on test instructions.

Finally, users were asked to retrieve those annotations for different targets and their higher parents' annotations.

The following describes some tasks in more detail. For example, KAU lab 129 was annotated with an annotation saying "This lab is reserved for Dr. Sameer", the context time associated with it was Monday from 1pm–4pm. The user groups who can retrieve this annotation is set to be "general", so that anyone can access it when retrieving annotations on lab 129 on Monday from 1pm–4pm only; at other times, this annotation wouldn't be available. Also, the same lab was annotated with another annotation saying "the quiz is this Monday". The context time was also Monday from 1pm to 4pm; however, the user's category is set only for group 7, so that group 7 users only can retrieve and access this annotation.

Also, in the lab 129, one of the computers (or PCs) and also the lab were annotated with annotations such as saying "PC14 needs to be fixed", where the context was all time. The user's category was "Technical support", so any staff of the technical support will get access to this annotation (we assumed one of the participants works in technical support).

Another student also annotated the lab with general annotations, such as one saying "I lost my wallet here; please, if you find it, hand it to the CS receptionists". The context was for a period of a week for everyone, so that anyone can retrieve that annotation for the following week after that annotation was created.

Another task that was given to participants is to annotate outdoor areas. For example, the entrance of the Computer Science department was annotated with different annotations. One was for new students' category saying "in this week the orientation session will be in theater 279 on Tuesday from 9am to 11:30am", so that the new students will get access to this annotation when they are at the entrance of the building. Also another annotation was made for group 7's students saying "here is our meeting point" and context time was Wednesday 1pm to 1:20pm. Also, participants were asked to retrieve annotations for higher (relative to their location) level entities in a DPVW-model. For example, in lab 129, they were asked to check annotations for level 2 that contains the lab, and also the whole building of their faculty, in order to test one aspect of using the hierarchical model. Also, participants were asked to leave annotations for private use, where only themselves can access those annotations, such as leaving an annotation as a reminder for returning a book to the library – the user only will get the reminder when retrieving the annotation of that book. Therefore, participants were given different tasks to create or retrieve different annotations and using different contexts in order to understand the system and test the functionality and efficiency of using ALPHAsys as a PA system.

Table 3
Results of users' experiences with ALPHAsys

	KAU	La Trobe	AVG
Q1	10	9.4	9.7
Q2	6	8	7
Q3	6.6	8.8	7.7
Q4	8	8.4	8.2
Q5	6	7	6.5
Q6	7.4	8.8	8.1
Q7	6	8	7
Q8	6.5	8	7.25
Q9	7.2	8	7.6
Q10	6.6	8.2	7.4
Q11	7.6	8.8	8.2
Q12	5.4	7.8	6.6
Q13	8.8	9	8.9
Q14	7.6	9.2	8.4
Q15	8.2	9	8.6

We also install a similar system that is used for Augmented Reality and Physical Annotations. We chose to compare the system with the layar system (www.layar.eu).

After testing the system, participants were given a small survey to be answered. The survey has different questions including user's technology background; the participants have to answer each question from 1 for the lowest rating (or least agreement or ease of use), to 10 for the highest rating (or most agreement or ease of use). The main tasks include the following:

Q1 Familiar with and used smart phones.
Q2 Familiar with and used location based services (LBS).
Q3 The system is easy to use.
Q4 Attach annotations to single entities.
Q5 Attach annotations to a collection.
Q6 Retrieve annotations of single entities.
Q7 Retrieve annotations of a collection.
Q8 Retrieve annotations of a virtual group.
Q9 Showing annotations based on my interest.
Q10 Showing annotations based on my locations.
Q11 Showing annotations based on current contexts.
Q12 Using image recognition to retrieve or leave annotations.
Q13 Ability to access annotations of parents of locations.
Q14 Adding annotations to rooms.
Q15 Moving entities to another parent (i.e. updating the DPVW-model).

Table 3 shows the results obtained from participants in both KAU and LTU. The first question shows both groups are using smart phones enormously, which means they are familiar with the current mobile technologies and they depend on their phones widely. Then, question 2's answers are indicative of their comfort level with the use of location-based services. It is obvious that participants in LTU use LBS more than participants in KAU, due to the lack of apps that support GPS in Saudi Arabia. So, this justifies the different results for some questions between the two groups. Q3 is about the overall impression of using ALPHAsys.

Then, Q4 is about attaching annotations to single entities, and Q5 about attaching annotations to a collection. The result shows that annotating a single entity is easier than annotating a collection, this because in a collection, the user has to scan more than one time for each entity, whereas its one time for the single entity. Q6, Q7 and Q8 are about retrieving annotations without context. Q9, Q10 and Q11 examine annotation retrieval with users' current context and interest. Q12 evaluates the image recognition technique for leaving and retrieving annotations. Q13 to Q15 are about the annotated entities in the DPVW-model. As can be observed, users can easily add and adjust entities in the DPVW-model. In general, Table 3 also indicates that the system is easy and useful for participants whether for leaving annotations or retrieving annotations.

Some limitations of the test are that annotating collection takes longer time than annotating a single entity, since users have to scan each one individually. Using image recognition for creating or retrieving annotations is not working perfectly yet due to lack of accuracy in current image recognition APIs that are available on the market, however, it is working in the "ideal environment" such as the correct light level and the camera angle.

Feature comparison with Layar In our literature survey [3] we made a comparison between 14 systems that have been used for physical annotations. In this section, ALPHAsys features have been compared with Layar, which is one of most popular system for AR and physical annotations. Table 4 shows the comparison between the two systems.

Both systems support using different layers of annotation contents, so that users can choose a layer of interested annotations for him/her such as tourism, and restaurant, so in ALPHAsys users can choose layers via user categories and annotation types. Also, users in both systems could be grouped into different categories, which will allow the authors to target a specific or a private category.

Table 4
Comparing ALPHAsys with LAYAR

Feature	ALPHAsys	LAYAR
Has different layers	✓	✓
Has different groups of users	✓	✓
User can choose particular author or annotation categories	✓	✓
Support different contexts (location, time, date, nearby person, nearby object, entity dependency)	✓	×
Targets have clear types	✓	✓
Each annotation has a unique ID (usable by users)	✓	×
Support mapping property	✓	×
Uses dynamic physical and virtual world models	✓	×
Support the hierarchy of targets	✓	×
Support collection of targets	✓	×
Support virtual group target	✓	×
Support interference detection and resolution [4]	✓	×
Support collision detection and resolving [4]	✓	×
Support image recognition	✓	✓
Support for annotating indoor and outdoor objects	✓	✓
Support private, shared and public annotations	✓	✓

Context dependency is supported in both systems. However, context dependency in LAYAR is limited to time, date, and location. Whereas, ALPHAsys supports contexts genereally such as nearby person, entity and more as was discussed in Section 2.

In Layar, an annotated entity comes as one single entity. They define it only as 'geo' to represent a location, and 'vision' for image recognition. Besides the geo location and image recognition, ALPHAsys clarifies the kind of annotated entities.

In Layar, there is a table called POI to store annotated entities data, and there is a POIAction table to store actions for that POI. In ALPHAsys annotations are fully independent and stored in different tables from the annotated entities and each annotation has a unique ID. This allows the system to support the mapping property and use the same annotations for different targets by using annotation IDs only.

Another key difference in ALPHAsys is that it supports the DPVW-model, so that annotated entities are structured in a hierarchical model which allows dynamic features of leaving and retrieving annotations using such structures, such as retrieving parent annotations, more details were discussed in Section 3, whereas the Layar system doesn't support such a model.

A target in Layar comes as a single entity, whereas ALPHAsys supports single and collection targets, so that the user can annotate a collection or composite target with one annotation.

Virtual group is another feature that is not supported by Layar, but is supported by ALPHAsys. Users can create and annotate a virtual group by one annotation even if the constituents of the virtual group are at different locations. This feature helps users to avoid repetitive actions in labeling many objects with the same annotation and allows sharing annotations among different entities. For image recognition, both systems support detecting annotated entities by using image recognition. Layar, in its last version, started to support it and calls it Layar Vision. Also, both systems support outdoor and indoor annotations. Moreover, both systems support sharing annotations with others, or keeping them private.

Also, the ALPHA model can be used to represent the key aspects of Layar and so is general, though ALPHAsys attempts to support more extensive features in ALPHA. It must also be noted that Layar has particular user features that ALPHAsys does not support given its extensive history of development, which are not modeled within ALPHA.

7. Related work

There have been many augmented reality and physical annotation systems as reviewed in [3]. Those systems can be used for different purposes such as education [21,23], health [6,19], and tourism [22,29]. However, a formal approach to describing the essential aspects of such systems can be useful, as well as the need to develop a model of the physical world to support such systems. Spatial location models have been researched in the last decade, and many surveys have reviewed the development of location models, such as [5]. Good examples of world models or location models are the Nexus platform [10], and the semantic space by Microsoft [7]. Nexus [9,10] is one of the most common models that aims to provide a generic platform for ubiquitous computing applications which can be used for indoor or outdoor environments; it includes the physical world and the virtual objects, which means any external digital resources. The main location based models can be categorized into three main groups [1,5]: geometric location models, symbolic location models (hierarchal, topological, descriptive) such as those in

[11,13,16,17] which describe spatial relationships between entities (an example of this type of model is when you describe a room in a floor in a particular building[2]), and semi-symbolic or hybrid models [8,12]. We have used symbolic models for indoor annotations and geometric GPS coordinates for outdoor annotations. The DPVW-model, which can include models of outdoor locations and buildings, serves not only as a design artifact for a PA system (in general) but can be used by users to browse for annotations (by browsing from their current locations, automatically detected, to surrounding spaces via the location model). The location models also serve as a means to query for annotations and as an organizing principle for physical annotations.

8. Conclusion

In this paper, we have provided a theoretical conceptual model for the PA systems – we call it ALPHA which provides a foundation on which to build a PA system. Then, we defined and categorized different types of annotated entities which are singles, collections and virtual groups. This categorization allows users to get different type of annotations based on target types. Then, on top of ALPHA, we provided DPVW-models which aim to represent the physical world so that we can use the structure of the physical entities and their physical relationships for annotations, and also the virtual model (VM) (or domain model) for the annotated entities to describe the virtual entities or the virtual relationship between targets. Then, we discuss the different relationships that we can get between targets and annotations, i.e., seven types of them. Finally, in order to prove the usability of the system, we implemented it and tested it in a real environment with users, and also compared it with other PA systems to clarify the new concepts of ALPHAsys.

We also found that the users' evaluations of ALPHAsys were generally positive towards its functions and its features for annotating the physical world among us, whether for annotating objects, spaces or locations. Or even for annotating a single entity, a collection, or a virtual group of entities. We found that ALPHAsys can structure annotations of physical entities via the DPVW-model, which also gives users the ability to access/browse annotations for higher levels or even in lower level for any entity (according to the particular DPVW-model, which is application-specific). Also, the test shows ALPHAsys has new context-aware features that enrich the task of annotating the physical world.

Future work will include more features about detection and resolving interference and collision between annotations and annotated entities as already started in [4].

Moreover, in the near future, we aim to involve many students in a rural area to use ALPHAsys for educational purposes. This will include improving the user interface of ALPHAsys (to make it more attractive) and further simplifying approaches for PA tasks.

[2] The advantage of this model is that it is readable by humans and also will be more efficient to answer some queries such as find all rooms in the fifth floor, but the disadvantage of this model is the lack of the exact locations of the entities and also the need to compute the distance between two entities.

Acknowledgements

We would like to thank Nghi for her work on integrating image recognition via the mobile camera as a means of retrieving physical annotations.

References

[1] I. Afyouni, C. Ray and C. Claramunt, Spatial models for indoor and context-aware navigation systems: A survey, *The Journal of Spatial Information Science (JOSIS)* (4) (2013), 85–123.

[2] A.A. Alzahrani, S.W. Loke and H. Lu, A formal model for advanced physical annotations, in: *Proc. of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC)*, Sydney, Australia, 12–14 Dec. 2011, 2011, pp. 170–177.

[3] A.A. Alzahrani, S.W. Loke and H. Lu, A survey on internet-enabled physical annotation systems, *International Journal of Pervasive Computing and Communications* **7**(4) (2011), 293–315, doi:10.1108/17427371111189647.

[4] A.A. Alzahrani, S.W. Loke and H. Lu, Conflict and interference resolution for physical annotation systems, in: *Proc. of the12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, Melbourne, Australia, 2013, pp. 1331–1338, doi:0.1109/TrustCom.2013.159.

[5] C. Becker and F. Durr, On location models for ubiquitous computing, *Personal and Ubiquitous Computing* **9**(1) (2004), 20–31, doi:10.1007/s00779-004-0270-2.

[6] T. Blum, V. Kleeberger, C. Bichlmeier and N. Navab, Mirracle: An augmented reality magic mirror system for anatomy education, in: *Proc. of the Virtual Reality Short Papers and Posters (VRW)*, 4–8 March 2012, IEEE, 2012, pp. 115–116, doi:10.1109/vr.2012.6180909.

[7] B. Brumitt and S. Shafer, Topological world modeling using semantic spaces, in: *Proc. of the UbiComp 2001 Workshop on Location Modeling for Ubiquitous Computing*, 2001, pp. 55–61, doi:citeulike-article-id:1284596.

[8] S. Christoph and H. Dominik, Using semantic web technology for ubiquitous location and situation modeling, *Annals of GIS* **10**(2) (2004), 157–165, doi:10.1080/10824000409480667.
[9] D. Fritsch, D. Klinec and S. Volz, NEXUS – Positioning and data management concepts for location-aware applications, *Computers, Environment and Urban Systems* **25**(3) (2001), 279–291, doi:citeulike-article-id:6107622.
[10] F. Hohl, U. Kubach, A. Leonhardi, K. Rothermel and M. Schwehm, Next century challenges: Nexus – An open global infrastructure for spatial-aware applications, in: *Proc. of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, Washington, United States, ACM, 1999, pp. 249–255, doi:10.1145/313451.313549, 313549.
[11] H. Höpfner and M. Schirmer, Towards modeling locations as poly-hierarchies, in: *Proc. of the 24th GI-Workshop on Foundations of Databases "Grundlagen von Datenbanken 2012"*, Germany, May 29 – June 01, 2012, Vol. 850, CEUR-WS.org.
[12] C. Jiang and P. Steenkiste, A hybrid location model with a computable location identifier for ubiquitous computing, in: *Proc. of the 4th International Conference on Ubiquitous Computing*, Sweden, Springer-Verlag, 2002, pp. 246–263, 741480.
[13] P. Jin, L. Zhang, J. Zhao, L. Zhao and L. Yue, Semantics and modeling of indoor moving objects, *IJMUE: International Journal of Indoor Moving Objects* **7**(2) (2012), 153–158.
[14] Junaio, Available from http://www.junaio.com (accessed 7 June 2011).
[15] LAYAR, Available from http://www.layar.com (accessed 15 September 2010).
[16] D. Li and D.L. Lee, A lattice-based semantic location model for indoor navigation, in: *Proc. of the the Ninth International Conference on Mobile Data Management*, IEEE Computer Society, 2008, pp. 17–24, doi:10.1109/mdm.2008.11, 1397836.
[17] B. Lorenz, H.J. Ohlbach and E.-P. Stoffel, A hybrid spatial model for representing indoor environments, in: *Proc. of the 6th International Conference on Web and Wireless Geographical Information Systems*, Hong Kong, China, Springer-Verlag, 2006, pp. 102–112, doi:10.1007/11935148_10, 2080202.
[18] K. Nakayama, T. Maekawa, H. Tomiyasu, T. Hara and S. Nishio, GeoNote.net: A social network system for geographic information, in: *Proc. of the 7th International Conference on Mobile Data Management (MDM'06)*, 2006, pp. 144–144.
[19] C. Orwat, A. Rashid, C. Holtmann, M. Wolk, M. Scheermesser, H. Kosow and A. Graefe, Adopting pervasive computing for routine use in healthcare, *IEEE Pervasive Computing* **9**(2) (2010), 64–71.
[20] P. Persson, F. Espinoza, P. Fagerberg, A. Sandin and R. Qoster, GeoNotes: A location-based information system for public spaces, in: *Designing Information Spaces: The Social Navigation Approach*, Springer-Verlag, 2003, pp. 151–173.
[21] E. Redondo, A.S. Riera, D. Fonseca and A. Peredo, Architectural geo-e-learning, in: *Virtual, Augmented and Mixed Reality. Systems and Applications*, Springer, 2013, pp. 188–197.
[22] M.C. Rodriguez-Sanchez, J. Martinez-Romo, S. Borromeo and J.A. Hernandez-Tamames, GAT: Platform for automatic context-aware mobile services for m-tourism, *Expert Systems with Applications* **40**(10) (2013), 4154–4163, doi:http://dx.doi.org/10.1016/j.eswa.2013.01.031.
[23] C. Van Aart, B. Wielinga and W.R. Van Hage, Mobile cultural heritage guide: Location-aware semantic search, in: *Knowledge Engineering and Management by the Masses*, Springer, 2010, pp. 257–271.
[24] W. Viana, J.B. Filho, J. Gensel, M.V. Oliver and H. Martin, PhotoMap – Automatic spatiotemporal annotation for mobile photos, in: *Proc. of the 7th international conference on Web and Wireless Geographical Information Systems*, Cardiff, UK, Springer-Verlag, 2007, pp. 187–201.
[25] Vuforia software platform, Available from: https://developer.vuforia.com (accessed 12 June 2012).
[26] K. Weigelt, M. Hambsch, G. Karacs, T. Zillger and A.C. Hubler, Labeling the world: Tagging mass products with printing processes, *IEEE Pervasive Computing* **9**(2) (2010), 59–63.
[27] Wikitude World Browser, Available from http://www.wikitude.org (accessd 1 September 2010).
[28] J. Wither, S. DiVerdi and T. Höllerer. Annotation in outdoor augmented reality, *Computers & Graphics* **33**(6) (2009), 679–689.
[29] W.-S. Yang and S.-Y. Hwang, iTravel: A recommender system in mobile peer-to-peer environment, *Journal of Systems and Software* **86**(1) (2013), 12–20, doi:http://dx.doi.org/10.1016/j.jss.2012.06.041
[30] YELLOW ARROW, Available from: http://yellowarrow.net/v3/index.php (accessed 8 June 2009).