Data Management Strategies for Mobile Augmented Reality

Gerhard Reitmayr and Dieter Schmalstieg
Vienna University of Technology
{reitmayr|schmalstieg}@ims.tuwien.ac.at

***Abstract*— Any significant real-world application of mobile augmented reality will require a large model of location-bound data. While it may appear that a natural approach is to develop application-specific data formats and management strategies, we have found that such an approach actually prevents reuse of the data and ultimately produces additional complexity in developing the application. In contrast we describe a three-tier architecture to manage a common data model for a set of applications. It is inspired by current Internet application frameworks and consists of a central storage layer using a common data model, a transformation layer responsible for filtering and adapting the data to the requirements of a particular applications on request, and finally of the applications itself. We demonstrate our architecture in a scenario consisting of two multi-user capable mobile AR applications for collaborative navigation and annotation in a city environment.**

Keywords: Augmented Reality, Mobile Computing, XML, GIS

I. Introduction

Many researchers believe that Augmented Reality (AR) is an excellent user interface for mobile computing applications, because it allows natural information browsing of location referenced information. The information is provided by a ubiquitous computing infrastructure and displayed using AR techniques.

Such applications require a detailed model of the user's environment. The model includes semantic and contextual elements related to a (potentially dynamic) real environment. Therefore AR models are more difficult to produce and maintain than typical virtual reality and visual simulation applications, which concentrate on visual fidelity. Detailed information on the environment is often available in legacy systems and needs to be extracted and transformed to be useful for the AR application.

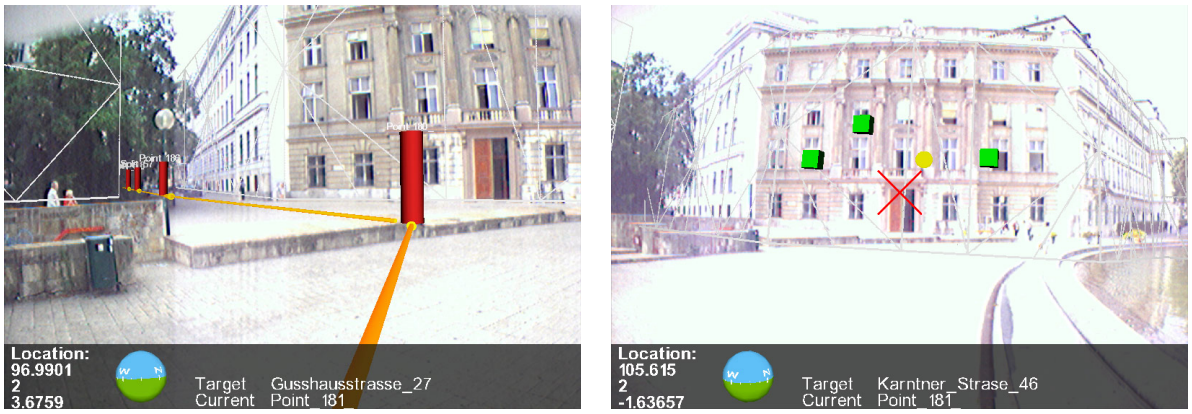
However, AR not only requires integration of a wider variety of data sources to build interesting applications, it also creates new types of content. For example, in the virtual showcase project [1], light maps are used to provide detailed lighting effects on real objects. In AR, geometrical models of real objects are frequently not used for visualization purposes, but for dealing with computation of occlusion, rendering of shadows, interaction, and vision-based tracking of real objects.

In a ubiquitous computing environment, multiple applications and users need to share the same environment model. These applications should be based on a common database which should also be capable of shared access for modification and updates. Since different applications will potentially not work with the same abstraction or representation of the model data, it may be difficult to keep the model data consistent if changes cannot be uniquely traced back to the data source.

Fig. 1
A mobile user roaming historical locations in the city of Vienna. He is equipped with a see-through HMD, a video camera, inertial tracker and GPS receiver mounted to a helmet. The mobile AR system is mounted to a backpack.

To address the complex modelling and data handling needs of ubiquitous augmented reality applications, we present the concept of an n-tier application architecture based on XML [2] as the enabling technology. A central XML-based database stores a common model described in section III used for all applications. Using common XML tools, data is transformed and imported from different sources. Once in the database, the data can be maintained more easily and application or domain specific preprocessing operations can be applied. At run time, the client applications can query the central store for relevant pieces of information (e.g. based on the current location). Before the data is delivered to the client, it can be transformed from its original form to a form directly useful to the client application. These transformations will often cull the model to return only those aspects of the data relevant to the application.

In this work, we aim to demonstrate how to set up a tool chain that can be used to develop compelling applications that scale well with the size of the model being used. We have investigated this approach during the development of a collaborative outdoor augmented reality application for navigation, information browsing, and annotation (see Fig. 1) which benefits directly from the proposed architecture and



a) b)

Fig. 2

A) THE NAVIGATION APPLICATION SHOWS THE WAY POINTS AND CONNECTING PATHS TO THE USER. B) A USER CAN SET ANNOTATION ICONS BY CONTROLLING A RAY CAST INTO THE SCENE. BOTH IMAGES ARE SCREEN SHOTS OF THE MOBILE SETUP IN VIDEO SEE-THROUGH MODE.

the used set of tools. While we have a working prototype and encouraging results, not all concepts proposed in this paper have been fully implemented yet. We feel that a number of questions require further investigation before they can be addressed.

A. Related Work

Navigation and information browsing are two common themes used for demonstrating wearable computing and mobile augmented reality. The Touring Machine [3] and wearable navigation applications described by Thomas et al. [4] and the Context Compass [5] show how pedestrian navigation can benefit from heads-up displays and AR interfaces. Information browsing was first demonstrated by the Navicam [6] and has since become a constant topic of AR applications. A successful example of a wearable tourist guide application is the GUIDE project [7] that presents roaming tourists with location based information on Lancaster. Finally, the GEIST [8] project provides historical information in an AR interface.

Typical AR demonstrations work with small data sets that have been entered manually and do not require data warehousing. To our knowledge, there has been little work done on data management techniques for large AR models. One piece related work by Julier et al. [9] addresses the issue of selecting appropriate data for display, from a user's point of view rather than that of the application. Höllerer et. al [10] describe the use of a database and description logic to store a model of a building floor which is annotated with meta-data for navigation target selection. The sentient computing project [11] uses a CORBA run-time infrastructure to model a live environment as distributed software objects where locations and attributes of objects are updated permanently. Newman et al. [12] describe a set of AR applications based on this infrastructure.

II. APPLICATION

We start by describing our current navigation and information browsing application for the city of Vienna as an example scenario for an augmented reality application that integrates a large amount of data from different sources. It provides a navigation aid that directs the user to a target location and an information browser that displays location referenced information icons that can be selected to present more detailed information in a variety of formats. Both functions support collaboration between multiple mobile users.

A. Navigation application

In navigation mode the user selects a specific target address or a desired target location of a certain type such as a supermarket or a pharmacy. The system then computes the shortest path in a known network of possible routes and displays world-stabilized way points and guidance arrows to the user (see Fig. 2 a). It is interactive and constantly reacts on the user's actual movements. It continuously recomputes the shortest path to the target if the user goes astray or decides to take another route.

If two or more users are present, a number of collaborative interactions are possible. A user can decide to always follow another user, or to set a target point for another user, or to meet halfway with another user. In the latter case the meeting point is calculated by the navigation system, but can be changed to a more suitable location if desired. Then both users will be guided to the meeting point.

B. Annotation application

In annotation mode the user selects a number of topics she is interested in. Location referenced information icons appear in her view. By using a virtual ray cast tool she can select certain topics to view the information content associated with the icons. This is used to convey historical and cultural information about sights in the city of Vienna. In addition to pure browsing, users can also annotate the model by placing virtual markers of different shapes and colors on structures and buildings in their surroundings (see Fig. 2 b).

The information browsing mode supports multiple users. Users can choose to share their selection of topics, or alter-

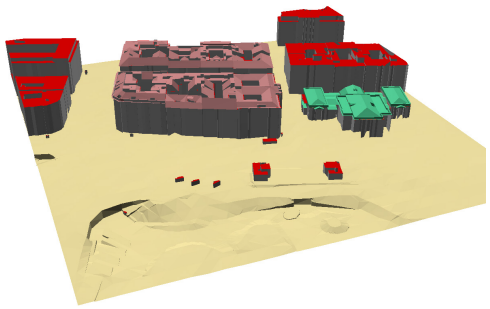


Fig. 3
A SUBSET OF THE 3D MODEL OF THE CITY OF VIENNA. IT INCLUDES A DIGITAL ELEVATION MODEL, BUILDING BLOCKS AND ROOFS. 3D MODEL CURTESY OF VIENNA CITY ADMINISTRATION.

natively, a tour guide can remote control the selection for a group of guided users. A user can also trigger the highlighted information on another user's display. The virtual markers are shared information and can help to point out features on distant structures such as building facades. Users can attach and discern different meanings associated with markers by assigning different styles.

C. Mobile system

These two applications were implemented on a mobile AR system [13]. Our current setup uses a notebook computer with a 2GHz processor and an NVidia Quadro4 graphics accelerator operating under Windows XP. It includes a wireless LAN network adapter to the notebook to enable communication with a second mobile unit. A Trimble Pathfinder Pocket differential GPS receiver is used to determine the position of the system in outdoor applications. All the equipment is mounted to a backpack worn by the user. We use a Sony Glasstron see-through stereoscopic color HMD fixed to a helmet as an output device. An InterSense InertiaCube2 orientation sensor and a PointGrey Research Firefly camera for fiducial tracking are mounted on the helmet (see Fig. 1).

We use Studierstube [14] which based on Open Inventor (OIV) [15] as our AR software platform. It provides a multi-user, multi-application environment, and supports a variety of display devices including stereoscopic HMDs. It also provides the means of 6DOF interaction, either with virtual objects or with user interface elements displayed in the user's field of view. Applications are developed as scene graphs that can be described with the declarative OIV file format. Studierstube is a very capable rapid prototyping system, but does not incorporate any database functions beyond a scene-graph-based run-time data structure.

D. Data aquisition

The scenario described above requires diverse data from a variety of sources. To integrate the data, we define a common data model that incorporates all features necessary for our application. This model and its applications are described in more detail in section III, while the remainder of this section focuses on data sources.

A 3D model of a part of Vienna was obtained from the cartography department of the city administration in 3D Studio Max format (see Fig. 3). This model was created as a reference model of Vienna, and is part of the official general map of the city [16]. The department of Geoinformatics at Vienna University of Technology supplied us with a network of accessible routes for pedestrians, delivered in GML2 format [17], an XML based file format used in the geographic information systems (GIS) community. This model was derived from the general map of Vienna and is represented as an undirected graph. Each node in this graph is geo-referenced and used as a way point in the navigation model. For each building, a so-called address point was defined and included into the path network to be able to construct a path to this address.

Furthermore, annotation information such as businesses located at certain addresses was derived from the general map of Vienna. This information is connected to address points in the spatial database. Cultural information taken from a guide book was included at various places to provide more detailed data for a tourist application. Finally, we placed the icons as spatial representations of this information into our model.

III. DATA MODEL

The architectural concept is based on a typical 3-tier model (see Fig. 4). The first tier is a central database storing the overall model. The second tier mediates between database and application by converting the general model from the database into data structures native to the respective application. The applications themselves are the third tier and only deal with data structures in their own native format.

This architecture provides the usual advantages of the n-tier model. A common data model and store reduces the amount of redundancy in the data required for different applications and allows centralized and efficient management of this data. The middle layer separates the presentation issues from the actual data storage and the applications. Thus the applications can be developed using efficient data structures independent of the actual storage format. Moreover, the transformation can be adapted to changing data formats or processes without touching either the application or the storage back-end, because it is a distinct entity separated from both.

We propose to use such an architecture and build upon XML technology, leveraging recent developments in the web development community. The proposed architecture is very common in this area and directly supported in a number of products, either open-sourced or developed commercially. The use of XML has a number of advantages for our task:

- A hierarchical data model fits well to our general spatial model. Rather than using a flat enumeration of building representations, a hierarchical model can represent several levels of a spatial hierarchy, from districts and streets down to rooms within buildings and other detailed geometrical data.
- While a document and file-oriented approach is generally sufficient for research prototyping, it obviously lacks scalability. More powerful storage solutions are required

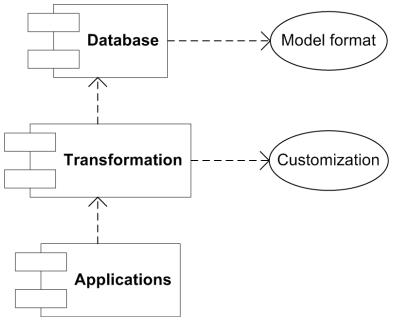


Fig. 4

DIAGRAM OF THE PROPOSED ARCHITECTURE. THE DATABASE COMPONENT STORES DATA IN THE MODEL FORMAT. TRANSFORMATIONS MEDIATE BETWEEN THE APPLICATIONS AND THE DATABASE AND CAN BE CUSTOMIZED TO FIT DIFFERENT OUTPUT REQUIREMENTS.

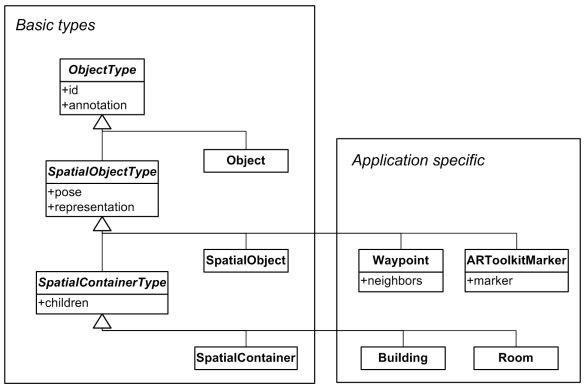


Fig. 5

OVERVIEW OF THE TYPE HIERARCHY IN THE MODEL FORMAT. A SET OF BASIC TYPES CAN BE USED FOR GENERAL MODELLING. APPLICATIONS MAY DERIVE ADDITIONAL TYPES FOR SPECIFIC REQUIREMENTS.

for real applications. Some of these exist in the form of XML databases. As XML technology is generally aimed at compatibility, the tools and APIs used for prototyping are directly supported by commercial XML products, and the transfer to a production system is greatly simplified.

- XML tools such as XSLT [18] allow rapid prototyping and development of import, transformation and export tools to and from the data model.
- Parsers and generators exist for a wide range of programming languages, and allow applications and tools to use the most appropriate language for the task.
- Standards for meaningful descriptions of data exist, on a syntactic level such as RDF [19] as well as on a semantic level for meta-data such as the Dublin Core [20]. This allows to define and use ontologies to support semantically rich queries and interactions.

We will describe each of these features in more detail by using the example of the tourist application described above.

A. Modelling

At the heart of our architecture lies a data model that encompasses the requirements of all applications. Care was taken in keeping the model extensible so that new data could be integrated during the development. This data model is described by an XML schema.

The model is based on an object-oriented approach using a type hierarchy of entities. The root type is called *ObjectType* and contains an id and a generic annotation subelement that can be used to store any XML tree. All data types defined in the model are derived from this type. The *SpatialObjectType* adds pose information and a geometrical representation to the super class. We further derive the *SpatialContainerType* that adds a children subelement to aggregate entities of type *ObjectType* for hierarchical composition.

From the three base types, we derive a number of application specific types that are used in the actual data files. The *Object*, *SpatialObject* and *SpatialContainer* elements are used for general purpose data and correspond directly to the base types (see Fig. 5). Applications can define additional types derived from the base types to provide more specific information. For example, the *Waypoint* element has a specific subelement to define neighboring waypoints connected by a path. Because elements refer back to their base type, an application can always provide a reasonable fallback behavior if it encounters an unknown derived application elements.

The XML tree is interpreted in the standard geometrical way, by defining a child's pose relative to its parent. We chose this mapping to support conventional modelling of visual data as directed acyclic graphs or trees. However, the open XML based format is not bound to any particular visualization tool or platform, and affords the definition of other than spatial relations by using relational techniques such as referring to object ids.

The annotation subelement of the abstract root type can be used to model free form data or to augment pre-existing types with extra information. This allows us to use more flexible technologies to annotate the objects in our model. For example, the information icons are modelled as simple *SpatialObjects* that are annotated with keywords and content. Different representations are generated for the annotation application based on a mime type stored in the content. Similarly we could use any meta-data standard to annotate our data.

B. Data handling

Having defined a model and data format, there are a number of tasks and tools necessary to (1) fill the database, (2) transform and manipulate the data and (3) finally make it available to the user through the development of appropriate applications. The typical tasks include the following:

(1) Import. Extract information from source data formats based on XML or other formats and map it to the data model. Non-XML source formats also require the combination of an appropriate parser with an XML generator to map the foreign format to XML.

Within our navigation application we need to convert data from a 3D file format to a subset of our XML format relating to geometrical descriptions. The navigation graph data was

available in GML2, therefore a simple XSLT transformation script was sufficient to incorporate this data into the model. Finally some data needed to be authored by hand as it was taken from a guide book.

(2) Maintenance. Maintaining a model requires the application of filters and transformations on data stored in the model format. It was necessary to compute the intersection of the 3D model data and the navigation graph, as the relevant input data was derived from two overlapping sections of the city map. This was achieved by computing a subset of the model within a given bounding box and then repairing the internal structures of the navigation graph to make sure the data is still coherent after the trimming. The maintenance tool directly reads from and writes to the common data model.

(3) Export. In the last step transformations are applied to retrieve relevant application data from storage and generate data structures for the applications. As described in section II-C, applications are implemented as Open Inventor scene graphs. Each application uses a custom XML stylesheet to directly generate the required scene graph and additional data structures from the general model.

For example, the navigation application uses a graphical representation of all waypoints and connecting paths to be able to visualize these to the user. The stylesheet generates an appropriate scene graph that also contains control structures for graphical effects, such as switches for visibility of certain parts of the model, which makes it easy to only display the waypoints along the current path. Moreover, it also generates an abstract representation of the navigation graph as an edge list which is used by the application to compute a shortest path to the target location.

The use of a separate step to transform the data into the application format has a number of advantages. It separates the general data format from the application specific data structures and provides an interface between both. It also provides a point for customizing the presentation independently of the application, similar to the way traditional cascading stylesheets work for HTML. As the stylesheet generates the actual graphical content, it can adapt it to different output requirements or profiles.

C. Database Persistence

A complete scalable production system for mobile AR will require a server-based persistent database, which can be accessed by clients over a network. At this point, we cannot report on hands-on experience with such a database, since our current prototype system uses a much simpler file based approach and executes the transformations offline. However, we expect it to be straightforward to build a distributed version based on HTTP or other protocols.

The transformation layer can either be implemented at the data store or as part part of the client application that filters all communications with the data store through the transformation layers. An advanced design could split the transformations layer into functions running on the server such as filtering and selections and pure data structure generators at the client side, for optimal use of network bandwidth. We plan to investigate the development of such a service facility based on an XML database to store the model as XML fragments combined with a transformation engine and network access.

IV. Discussion

The proposed architecture and set of tools are based on established techniques in current computer science. However, we have found little evidence of using scalable data management techniques for large scale AR. There is a small set of related areas where indirectly applicable techniques are used: GIS databases are primarily aimed at 2D information and static, stationary use cases. Location-based services are aimed at abstract, text-oriented information and very low-end devices. Visual simulation focuses on high-fidelity graphical representation and stationary displays. In this work, we have attempted to explore the combination of aspects of these areas into a scalable mobile AR database system.

Today's most common technology for storing data are relational database management systems (RDMS). However, we did not base our approach on a relational data model for several reasons. The hierarchical structure inherent in 3D data models is more directly mapped to an XML structure. One of the great advantages of XML technology is its self-description and self-organisation through schemas and namespaces. AR applications benefit from this flexibility, because data definitions can be decoupled by domain and developed independently without breaking application compatibility.

In principle, these aspects can also be addressed with a relational model, but at the expense of imposing the organizational workload on the developer rather than the underlying tools. Nevertheless, an RDBMS provides a solid performance basis for implementing an XML storage system and will probably play an important part in any production system following the proposed architecture.

V. Open research questions

The work described so far is unfinished and there are a number of unanswered questions. An important question to be addressed in future research work is whether a common data model for a whole set of augmented reality applications is feasible. While it might not be desirable (or even possible) to define a common data model for all AR applications in general, it certainly appears worthwhile to investigate this approach for certain well-defined application domains. In the wearable AR domain, a model of the environment and associated information is a prerequisite to providing any interface beyond head-stabilized displays. Therefore, our data model can serve as a foundation for a wide area of AR applications.

How should such a model be defined and what are the best ways to do that? Our initial approach has been to define a static XML schema that was extended as needed. However, more advanced techniques such as plugging together different formats by embedding XML fragments with namespaces or describing model properties with ontologies that allow complex semantic queries are possible directions that need to be investigated.

Such concepts are actively developed and researched in the semantic web community [21], [22].

Modelling knowledge from other fields needs to be incorporated. The body of work from the Geoinformatics community surrounding GIS needs to be considered. While we have already incorporated GIS data into our system, we still need to research the model properties of GIS formats and identify common concepts and techniques. Parallel developments in the GIS community to incorporate 3D models [16] or using XML [23], demonstrate similar research interests.

Our current prototype only works offline with static files. In the next step, we plan to provide a distributed online service that allows mobile applications to retrieve the required data on the fly. Since mobile systems cannot be expected to be always connected in a uniform way, transport mechanisms, replication and caching as well as quality of service considerations will have to be investigated.

It is also interesting to examine how the proposed architecture can function as the basic substrate for a more intelligent infrastructure. The sentient computing system [11] implements a system of active objects that represent the state of the observed environment. These objects could be an active part of the environment model and modify it as required. Other systems such as DWARF [24] take a completely distributed approach in which an application is built from distributed software components. This introduces the idea of a distributed database to support a more loosely coupled system approach and requires dealing with synchronization and concurrent updates to different instances of the database.

VI. Conclusions

We presented an architecture for building data intensive augmented reality applications based on XML technology. The features of the architecture simplified development of data-intensive demonstration applications that scale with the amount of data. While the data management strategies presented here are established in other data-intensive domains, there is no precedent for the use in the field of AR, which has its own unique set of requirements and peculiarities. Therefore, a number of interesting research questions could be identified and we would like to further discuss and investigate building integrated world models for augmented reality applications.

Visit the Outdoor Collaborative Augmented Reality web site for more information on the tools which were used: http://www.ims.tuwien.ac.at/research/projects/ocar/

Acknowledgments

This work was sponsored by the Austrian Science Foundation *FWF* under contracts no. P14470 and Y193, and Vienna University of Technology by *Forschungsinfrastrukturvorhaben TUWP16/2002*. We would like to thank the Vienna City Administration for providing the 3D model of Vienna, Prof. Stephan Winter for the navigation model and Joe Newman for setting up the DGPS system.

References

[1] O. Bimber and B. Fröhlich, "Occlusion shadows: Using projected light to generate realistic occlusion effects for view-dependent optical see-through displays," in *Proc. ISMAR 2002*, (Darmstadt, Germany), pp. 186–195, ACM and IEEE, September 30 – October 1 2002.

[2] T. Bray, J. Paoli, C. M. Sperberg-McQueen, *et al.*, "Extensible markup language (XML) 1.0." http://www.w3.org/TR/REC-xml/.

[3] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster, "A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban enviroment," in *Proc. ISWC'97*, (Cambridge, MA, USA), pp. 74–81, October 13–14 1997.

[4] B. H. Thomas, V. Demczuk, W. Piekarski, D. H. epworth, and B. Gunther, "A wearable computer system with augmented reality to support terrestrial navigation," in *Proc. ISWC'98*, (Pittsburgh, USA), pp. 168–171, IEEE and ACM, October 19–20 1998.

[5] R. Suomela and J. Lehikoinen, "Context compass," in *Proc. ISWC 2000*, (Atlanta, Georgia, USA), pp. 147–154, IEEE, October 16–17 2000.

[6] J. Rekimoto, "Navicam: A magnifying glass approach to augmented reality," *PRESENCE - Teleoperators and Virtual Environments*, vol. 6, pp. 399–412, August 1997.

[7] N. Davies, K. Cheverest, K. Mitchell, and A. Friday, "'caches in the air': Disseminating tourist information in the guide system," in *Proc. WMCSA'99*, (New Orleans, LA, USA), IEEE, February 25–26 1999.

[8] U. Kretschmer, V. Coors, U. Spierling, D. Grasbon, K. Schneider, I. Rojas, and R. Malaka, "Meeting the spririt of history," in *Proc. VAST 2001*, (Glyfada, Athens, Greece), Eurographics, November 28–30 2001.

[9] S. Julier, M. Lanzagorta, Y. Baillot, L. Rosenblum, S. Feiner, and T. Höllerer, "Information filtering for mobile augmented reality," in *Proc. ISAR 2000*, (Munich, Germany), pp. 3–11, IEEE, October 5–6 2000.

[10] T. Höllerer, D. Hallaway, N. Tinna, and S. Feiner, "Steps toward accommodating variable position tracking accuracy in a mobile augmented reality system," in *Proc. AIMS'01*, (Seattle, WA, USA), pp. 31–37, Aug 4. 2001.

[11] M. Addlesee, R. Curwen, S. Hodges, A. Hopper, J. Newman, P. Steggles, and A. Ward, "A sentient computing system," *IEEE Computer: Location-Aware Computing*, August 2001.

[12] J. Newman, D. Ingram, and A. Hopper, "Augmented reality in a wide area sentient environment," in *Proc. ISAR 2001*, (New York, New York, USA), pp. 77–86, IEEE and ACM, October 29–30 2001.

[13] G. Reitmayr and D. Schmalstieg, "Mobile collaborative augmented reality," in *Proc. ISAR 2001*, (New York, New York, USA), pp. 114–123, IEEE, October 29–30 2001.

[14] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavari, L. M. Encarnao, M. Gervautz, and W. Purgathofer, "The Studierstube augmented reality project," *PRESENCE - Teleoperators and Virtual Environments*, vol. 11, no. 1, 2002.

[15] P. Strauss and R. Carey, "An object oriented 3D graphics toolkit," in *Proc. ACM SIGGRAPH'92*, ACM, 1992.

[16] L. Dorffner and A. Zöchling, "Das 3D modell von wien - erzeugung und Fortführung auf basis der wiener mehrzweckkarte," in *Proc. CORP 2003*, (Vienna, Austria), pp. 161 – 166, February 25 – March 1 2003.

[17] S. Cox, A. Cuthbert, R. Lake, and R. Martell, "Geography markup language (GML) 2.0." http://opengis.net/gml/01-029/GML2.html, 2001.

[18] J. Clark, "XSL transformations (XSLT) version 1.0." http://www.w3.org/TR/xslt, 1999.

[19] F. Manola and E. Miller, "Rdf primer." http://www.w3c.org/TR/rdf-primer/, May 3 2003.

[20] S. L. Weibel and T. Koch, "The dublin core metadata initiative: Mission, current activities, and future directions," *D-Lib Magazine*, vol. 6, December 2000.

[21] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, vol. 284, pp. 34–43, May 2001.

[22] "DARPA agent markup language (DAML) home page." http://www.daml.org/, July 30 2003.

[23] P. Dorninger, "XML technologies and geodata," in *Proc. CORP 2003*, (Vienna, Austria), pp. 223 – 229, February 25 – March 1 2003.

[24] M. Bauer, B. Bruegge, G. Klinker, A. MacWilliams, T. Reichner, S. Riss, C. Sandor, and M. Wagner, "Design of a component-based augmented reality framework," in *Proc. ISAR 2001*, (New York, New York, USA), pp. 45–54, IEEE and ACM, October 29–30 2001.