# Tracking for Distributed Mixed Reality Environments

Joseph Newman* University of Cambridge
Alexander Bornik† Technical University of Graz.
Daniel Pustka‡ Technische Universität München.
Florian Echtler§ Technische Universität München.
Manuel Huber¶ Technische Universität München.
Dieter Schmalstieg‖ Technical University of Graz.
Gudrun Klinker** Technische Universität München.

**ABSTRACT**

By developing a taxonomy of existing systems, libraries and frameworks used for developing Mixed Reality and Ubiquitous Computing environments, it can be seen that little effort has been made to research applications that share the characteristics of these two fields. Solutions have focussed on meeting the needs of each project in turn, rather than developing a generalised middleware, which would not only have the advantage of generality, but also facilitate cooperation and collaboration in what ought to be a cross-disciplinary area. We have designed just such a solution, that can be used to support a wide-range of Mixed Reality applications and allow them to inter-operate in a distributed fashion, whilst simultaneously supporting straightforward porting of legacy applications.

**Keywords:** Ubiquitous Computing, Pervasive Computing, Augmented Reality, Sensor Fusion.

**Index Terms:** D.2.11 [Software Engineering]: Software Architectures—Domain Specific Architectures; D.2.11 [Software Engineering]: Software Architecture—Patterns;

## 1 Introduction and Related Work

The field of Mixed Reality is not homogeneous, but highly diverse and cross-disciplinary. Whilst this field is now treated on its own merits in courses, conferences and journals rather than as sub-section of Virtual Reality, its practitioners come from fields as different as Wearable Computing, Ubiquitous Computing, Graphics and Computer Vision. In theory, these backgrounds ought to be complementary but there are barriers to mutual understanding and especially to collaboration, resulting from budgetary, technical and social constraints. In this paper, we examine some of the technical constraints using a new application taxonomy, and propose a new middleware framework which should reduce the overhead of practical cooperation.

## 2 Milgram-Weiser Continuum

Milgram proposed his Virtuality Continuum [5], shown in figure 2, which described the range of applications coming under the heading of Mixed Reality (MR) with the wholly virtual world of Virtual Reality (VR) and the entirely real world we ordinarily inhabit as extrema. There is unrealised potential in the coupling of these applications with the rich environments anticipated in Weiser's "third wave" [16] — computing capabilities embedded in even the most mundane objects through networks of numerous and cheap wireless sensors — enabling novel forms of visualisation and user interface. However, Weiser himself placed Ubiquitous Computing (Ubicomp) in opposition to Virtual Reality stating, "Ubiquitous Computing is roughly the opposite of virtual reality" [16]. When one considers that VR is merely at one extreme of the Milgram's continuum, then one can see that Ubicomp and VR are not strictly *opposite* one another but rather *orthogonal*. An analogous continuum, which we will posthumously call "Weiser's Continuum" (see figure 3), would have Ubicomp at one extreme and monolothic mainframe-based computing at the other. Placing these continua at right-angles gives us the 2D space shown in figure 1, in which different application domains represent areas in this space.

The gamut of Milgram's continuum runs from CAVE®s and tethered head-mounted display (HMD) setups, "mobile" backpack-based HMD setups such as the MARS system from Höllerer et al. [3], and rather more pocketable PDAs [15] and smartphones using a magic-lense metaphor. In its raw form a CAVE® or HMD would be the VR equivalent of an old-fashioned terminal connected to a mainframe: a conspicuous interface to the virtual world for a single lone user. This application therefore occupies the top-right hand corner of the Milgram-Weiser space. Similarly, a surgeon visualising a tumour using registered 3D graphical overlays in a specially designed AR-equipped operating theatre [14], is still interacting (quite appropriately in the context) with a monolithic computing architecture. Therefore, AR-assisted surgery is in the top-left of the Milgram-Weiser space. However, two CAVE®s in remote locations but placing their users in a *shared* space within a Distributed Virtual Environment [6] (DVE), encounter similar issues to those raised by Weiser relating to mobility within Ubicomp. Moving below the Reality-Virtuality axis in the Milgram-Weiser space towards the bottom-left of the diagram further Ubicomp concepts are introduced by Newman and Reitmayr in the form of "Wide Area Indoor Sentient AR" [7] and "Outdoor Collaborative AR" [12] [13] respectively. What these systems all have in common is a highly tuned mix of sensors, that have been chosen and combined in such a way as to meet specific application requirements: precise, high bandwidth, low latency data concerning a small number of predefined objects within a small volume. In contrast, Ubicomp systems have generally used numerous, cheap, low bandwidth and relatively inaccurate sensors scattered over wide areas, resulting in federated data models such as Nexus [1] and QoSDream [2].

Most studies into classic Mixed Reality or Ubicomp environments have typically picked single points within this 2D space, in order to investigate very specific properties of a given setup, or device. The result of this has been that solutions have been developed on a case-by-case basis, rather than aiming at a more general middleware enabling the development of systems encompassing the entire Milgram-Weiser space. Klinker et al. outlined the distributed tracking concepts [4] needed to exploit AR outside laboratory settings in the context of industrial processes, which lead to the formulation of just such a generalised middleware, which we call Ubiquitous Tracking [9] or Ubitrack. The original intention in developing this MR classification [8] was to identify and occupy a new and currently empty area in the bottom-left of the Milgram-Weiser space labelled "Ubiquitous AR". However, in the course

*e-mail: jfn20@cam.ac.uk
†e-mail:bornik@icg.tu-graz.ac.at
‡e-mail:pustka@in.tum.de
§e-mail:echtler@in.tum.de
¶e-mail:huberma@in.tum.de
‖e-mail:schmalstieg@icg.tu-graz.ac.at
**e-mail:klinker@in.tum.de

of the Presenccia [10] project, which requires distributed tracking facilities to co-exist and inter-operate in collaborative gaming applications across the entire Milgram-Weiser space, we are striving towards concepts that are sufficiently general to support tracking across the entire domain. This paper outlines a pragmatic design of such an architecture, conceived specifically with the aim of handling multiple paradigms and straightforward porting of legacy applications.

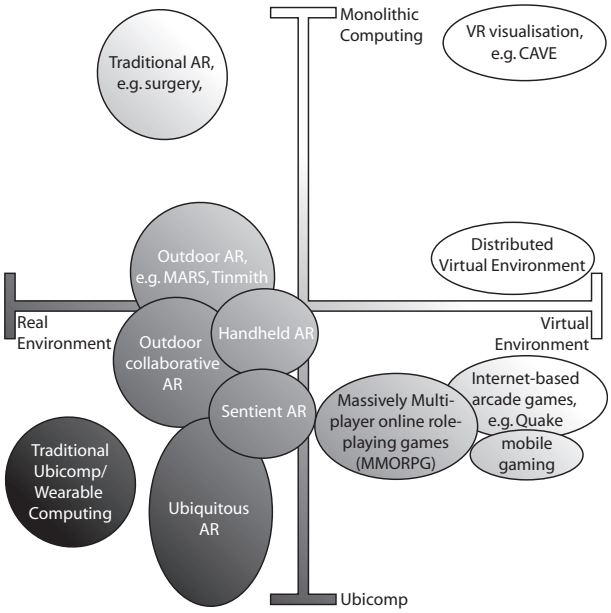


Figure 1: Milgram-Weiser Continuum

## 3 Ubitrack System Architecture

We propose a client-server-based middleware architecture for tracking which fulfils the requirements of the different scenarios in the Milgram-Weiser continuum. Figure 4 shows the basic components of the architecture and the most relevant communication channels. The core idea of the design is to separate the coordination and configuration of sensors and consumers from the actual generation, processing and consumption of sensor data. This data can be real or synthetic, corresponding to actual devices in the real world or virtual devices in a MR environment respectively. This separation permits the relatively resource-intensive coordination process to run on the server, while clients communicate directly with one another in accordance with the low-latency requirements of AR/VR applications. In this context, the Ubitrack architecture can be seen as a centrally coordinated peer-to-peer architecture, rather than a pure client-server solution.

### 3.1 Ubitrack Server

The Ubitrack Server is the central component of the system, which maintains a database of all objects and their relationships as a Relationship Graph (RG), which itself consists of a subgraph made up of coordinate frames, sensors and tracked objects together with their *spatial* relationships forming a Spatial Relationship Graph (SRG). This database, though initially empty, contains the aggregated spatial (and other) relationships specified by clients and thus contains complete knowledge about the topology of the tracking infrastructure. It does not, however, contain actual sensor measurements.

Clients can register queries concerning parts of the SRG, including spatial relationships that cannot be measured directly, but can only be inferred using Spatial Relationship Patterns [11]. The server constantly matches the SRG against registered client queries and, if a match is found, a data flow network description (see below) is generated and sent to the client. The server may also order a client to construct a data flow network that supports other clients by processing tracking data and transmitting them over the network, thus sharing trackers between clients.

### 3.2 Ubitrack Client Framework

The clients of a Ubitrack system can be sensors, rendering devices and other user-machine interaction components, but also mixed forms, such as virtual characters who both react to the environment and provide information about their location in space. Additionally, existing legacy applications can be implemented as a single client capable of consuming additional sensor data and/or to disclose information from contained sensors or internal state. Ubitrack clients are implemented using the Client Framework represented by the box in the bottom-left of figure 4.

**Ubitrack Client** In order to keep the interface for application programmers minimal, the client-side network communication is encapsulated in the Ubitrack Client library. It supports interaction with the Ubitrack system at various degrees of complexity, ranging from simple "where is object A" queries to persistent requests for tracking data of all objects matching a given predicates with additional specification of desired tracking quality and base coordinate frames.

**Application** A developer writes this component conforming to certain interfaces that govern how the application informs the Ubitrack Server via the Ubitrack Client of the tracking devices and objects for which it is responsible, and by which the Ubitrack Server can similarly inform the application of the objects in which it has expressed interest.

**Data Flow Network** The data flow network is the part of the client-side architecture responsible for generating and processing tracking events, and for sending them over the network. The emphasis is on high throughput with low latency and rapid reconfigurability to accommodate changes in the infrastructure. It consists of a pipes-and-filters architecture, as used by OpenTracker [13], with drivers for trackers and components for transformation and network transport. Additionally, `Callback` and `Callforward` components provide an interface through which the actual application can send and receive events.

The structure of the data flow network is determined at runtime by the server in response to an application's queries and the state of the tracking infrastructure. The (re-)configuration is done by the Ubitrack Client, which maintains the connection to the Ubitrack server.

### 3.3 Spatial Indexer

In many applications, clients are only interested in tracking objects which are contained in a particular region such as, for example, a room. However, this "containment" information cannot usually be determined from the topology of the Spatial Relationship Graph, but only by examining the sensor data itself. To avoid information overload at the Ubitrack Server, parts of the system that depend on sensor data to evaluate client queries are separated from the server itself using a plug-in mechanism. The range of predicates that clients can use to further restrict the results of a query (e.g. `containedIn`) is supported by the server by running a spatial indexing service, that otherwise behaves similarly to a normal Ubitrack client.

### 3.4 Communication Channels

To support communication between the different components of a Ubitrack system, standardized protocols will be necessary. We have

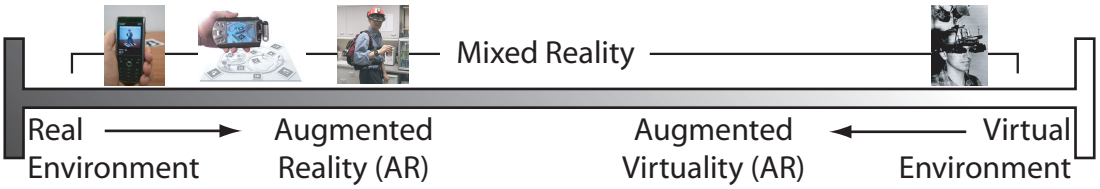


Figure 2: Milgram's Reality-Virtuality Continuum

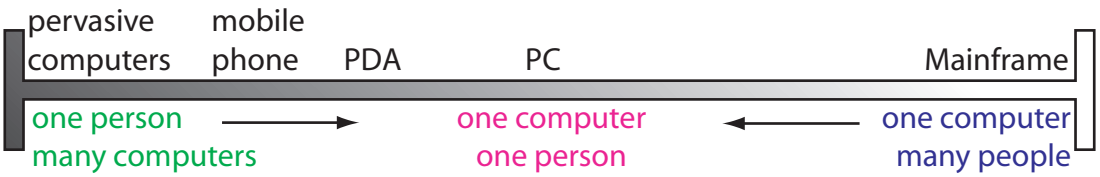


Figure 3: Spectrum ranging from Ubiquitous to Monolithic Computing posthumously "Weiser Continuum"

identified three communication channels, for which different protocols will have to be designed, due to their distinct natures.

Client–Server The interface between the client and the server is the core interface of the Ubitrack system. We envision that it will be an open standard, generic enough to allow interoperability between different client versions and data flow network implementations. Clients need to be able to specify their coordinate frames, sensors and tracked objects in the form of an SRG; they need to express queries for tracking data of single or multiple objects with optional quality constraints. Additionally, clients must be able to specify their networking and processing capabilities for interoperability of different data flow network versions and implementations.

In the opposite direction, Ubitrack Servers respond to queries by sending replies consisting of data flow network configurations back to the client. The protocol needs to support incremental updates bi-directionally, notifying both clients and the server of changes to the infrastructure at runtime.

Client–Client Clients communicate with each other by exchanging tracking data. This exchange is centrally coordinated by the Ubitrack Server which instructs clients to integrate networking components into their data flow networks. The exact format depends on the data flow network implementation. For example OpenTracker has a range of different network sources and sinks using different protocols (e.g. TCP/IP and UDP multicast).

Server–Spatial Indexer The integration of a spatial indexer requires a dedicated interface over which the indexer can plug into the Ubitrack Server. This requires the indexer to specify which additional predicates it provides and a query interface the server uses to retrieve the set of objects that currently fulfill a given predicate.

### 3.5 Example

A deployment scenario in an airport setting for the proposed ubitrack architecture is shown in Figure 5 consisting of four clients:

1. RFID tags embedded in every boarding card are handed out at the check-in desk. RFID tag readers are deployed at key locations throughout the airport. As a passenger proceeds through security and the lounge areas to their gate, every time they pass within range of a RFID tag reader their location, to the granularity of the range of the reader, is detected.
2. A control centre is equipped with many large screens, which are used to visualise the flow of passengers through the airport and as endpoints for the pan-tilt zoom cameras and wearable cameras deployed throughout the airport.
3. A security guard is equipped with a hands-free headset, a camera and a heads-up display. The guard is able to communicate bi-directionally with the control centre using the audio headset. The purpose of the camera is twofold: it can track natural features and fiducial markers in the environment in order to localise the guard, and secondly provide a live feed in places where fixed cameras are occluded. The heads-up display can be used to overlay registered 3D graphics over the guard's vision, allowing a suspect package identified by the control centre to be pin-pointed or an individual to be questioned.
4. A spatial indexer takes tracking data to perform spatial reasoning, deducing higher level knowledge, e.g. has passenger *A* gone through security, or whether they are in a particular shop. The spatial indexer requests tracking data from as many sensors as possible.

Upon startup the mobile client assisting the guard registers its camera as a tracker with the Ubitrack Server, by composing a query which is sent via the Ubitrack Client component. The data flow network of the mobile client is initially empty, but is populated in response to the data flow configuration returned by the Ubitrack server. The result of this is that the pose of the guard is transmitted over a wireless network. The control centre application expresses its interest in all objects, also via a query, resulting in the pose of the guard being received, via another data flow network, which allows the pan-tilt zoom cameras to keep the guard in view. This configuration is unlikely to remain valid indefinitely, so the Ubitrack Server continually revises the data flow networks on all the clients as queries change, passengers and guards come and go, cameras pass out of visual range and trackers are switched on and off.

There is a feedback loop between mobile clients and the control centre that is closed by the humans within the loop: agents acting as security guards and supervisors from the control centre. In addition to the classic AR interfaces we might devise for the guard, it is important to maintain the flow of human communication with the control centre. Given sufficient cameras the guard will remain within view of the fixed cameras which means that, for example, a supervisor can instruct a guard to "look left a bit" for a particular object. We envisage new interfaces that robustly enhance this sort of collaboration, allowing people to work together more effectively and safely, rather than replacing them with security robots.

## 4 Future Work

The current architecture is centralised in design, with a single Ubitrack Server that is responsible for coordinating all the tracking de-

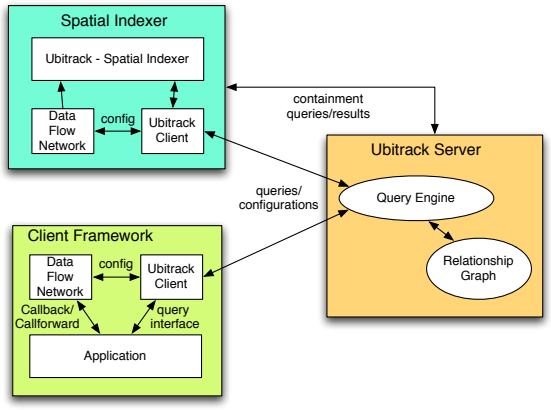


Figure 4: Ubiquitous Tracking Architecture

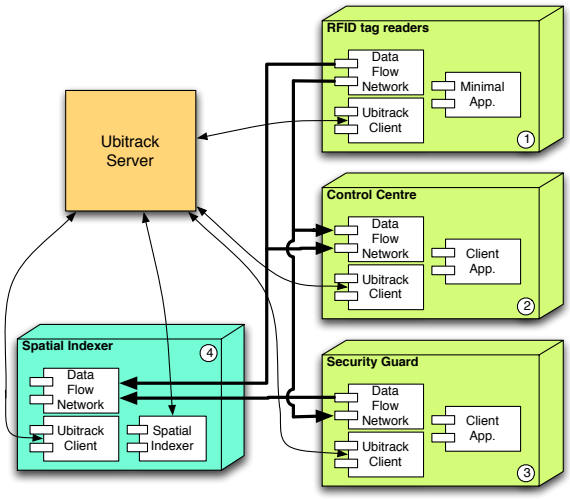


Figure 5: Example deployment

vices, sensors and applications in the environment for which it is responsible. Whilst we believe that this design will scale beyond the distributed MR environments we are currently able to build, at some point there will be an inevitable limit to the number of queries, devices and applications it can handle. We will explore alternative approaches whereby the responsibility for managing devices will be shared by many distributed Ubitrack Servers that will seamlessly hand off between one another, without the awareness of clients.

## 5 Conclusion

We have developed a taxonomy for classifying Mixed Reality and Ubiquitous Computing applications called the Milgram-Weiser continuum which can be visualised as a 2D space. The presence of gaps in interesting places, as well as a lack of an architecture for supporting inter-operability between the applications spread across this space, has motivated us to design a middleware architecture that can support the needs of applications from both fields and allow them to interact, whilst also providing support for legacy applications. It is anticipated that not only will this make such applications easier to develop and deploy, but it will stimulate cross-disciplinary endeavours in this area by reducing technical barriers to cooperation.

## Acknowledgements

This work was supported by the Bayerische Forschungsstiftung (project TrackFrame, AZ-653-05), the Austrian Science Fund FWF (contract no. Y193) and the PRESENCCIA Integrated Project funded under the European Sixth Framework Program, Future and Emerging Technologies (FET) (contract no. 27731).

## References

[1] M. Bauer and K. Rothermel. Towards the observation of spatial events in distributed location-aware systems. In *Towards the Observation of Spatial Events in Distributed Location-Aware Systems*, pages 581—582, Los Alamitos, California, July 2002. IEEE Computer Society.

[2] G. Coulouris. Review report: The qosdream project. Technical report, Laboratory for Communication Engineering, University of Cambridge, 2002.

[3] T. Höllerer, S. Feiner, T. Terauchi, G. Rashid, and D. Hallaway. Exploring mars: Developing indoor and outdoor user interfaces to a mobile augmented reality system. *Computers & Graphics*, 23(6):779–785, 1999.

[4] G. Klinker, T. Reicher, and B. Bruegge. Distributed user tracking concepts for augmented reality applications. In *Proc. of ISAR 2000, Munich*, Munich, Germany, Oct. 2000. IEEE Computer Society.

[5] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12), December 1994.

[6] J. Mortensen, V. Vinayagamoorthy, M. Slater, A. Steed, B. Lok, and M. Whitton. Collaboration in tele-immersive environments. In *Eighth Eurographics Workshop on Virtual Environments*, pages 093–101. Eurographics Association, 2002.

[7] J. Newman, D. Ingram, and A. Hopper. Augmented reality in a wide area sentient environment. In *Proc. of IEEE and ACM Int. Symp. on Augmented Reality (ISAR 2001)*, pages 77–86, New York, NY, October 2001.

[8] J. Newman, G. Schall, and D. Schmalstieg. Modelling and handling seams in wide-area sensor networks. In *Proc. of ISWC 2006*, October 2006.

[9] J. Newman, M. Wagner, M. Bauer, A. M. lliams, T. Pintaric, D. Beyer, D. Pustka, F. Strasser, D. Schmalstieg, and G. Klinker. Ubiquitous tracking for augmented reality. In *Proceedings of ISMAR 2004*, November 2004.

[10] Presenccia. —presence research encompassing sensory enhancement, neuroscience, cerebral-computer interfaces and applications. http://www.presenccia.org/ [2007, January 8].

[11] D. Pustka, M. Huber, M. Bauer, and G. Klinker. Spatial relationship patterns: Elements of reusable tracking and calibration systems. In *Proc. IEEE International Symposium on Mixed and Augmented Reality (ISMAR'06)*, October 2006.

[12] G. Reitmayr and D. Schmalstieg. Collaborative augmented reality for outdoor navigation and information browsing. In *Proc. Symposium Location Based Services and TeleCartography 2004*, Vienna, Austria, 2004.

[13] G. Reitmayr and D. Schmalstieg. Opentracker - a flexible software design for three-dimensional interaction. *Virtual Reality*, 9(1):79–92, December 2005.

[14] F. Sauer, A. Khamene, B. Bascle, and G. J. Rubino. A head-mounted display system for augmented reality image guidance: Towards clinical evaluation for imri-guided neurosurgery. In *MICCAI '01: Proceedings of the 4th International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 707–716, London, UK, 2001. Springer-Verlag.

[15] D. Wagner and D. Schmalstieg. A handheld augmented reality museum guide. In *Proceedings of IADIS International Conference on Mobile Learning 2005 (ML2005)*, June 2005.

[16] M. Weiser. Ubiquitous computing. http://www.ubiq.com/hypertext/weiser/UbiHome.html [2007, January 8], March 17 1996.