RoamBlog: Outdoor and Indoor Geo-blogging Enhanced with Contextual Service Provisioning for Mobile Internet Users

Vincenzo Pallotta, Amos Brocco, Dominique Guinard, Pascal Bruegger, Pedro de Almeida
Department of Computer Science
University of Fribourg, Switzerland
{Name.Surname@unifr.ch}

Abstract

RoamBlog is a mobile ubiquitous computing architecture that enables roaming internet users equipped with mobile networked devices such as Smartphones, PDAs and JavaPhones to continuously interact with geo-contextualized services. Within the RoamBlog architecture, we propose a novel user interface for mobile ubiquitous computing systems, the Kinetic User Interface (KUI), where physical motion along geographical locations is recognized as an “embodied” pointing device. We present a few usage scenarios for which applications have been designed on top of the RoamBlog architecture.

1 Introduction

Internet and mobile computer technology are changing the way users retrieve information and interact with media and services. *Personal computing* in its original form is disappearing and it is taking the shape of a global distributed system made of several interconnected heterogeneous computational components with different degrees of mobility and computing power. This paradigm-shift supports the vision of what is commonly called *ubiquitous computing* (Weiser, 1993).

As stated in (Weiser, 1991) there are two important issues concerning Ubiquitous Computing: *context* and *scale*. Context typically includes information about the actual usage situation of a computing device by a specific type of user in a distributed system, such as the spatio-temporal location of the device, the type of the device, the user and his/her preferences, and basically all additional information that can be captured by environmental sensors physically or logically connected to the device and its user. Scale refers to how suitable a given type of device is to carry out certain tasks, or better, what are the computational requirements and constraints to execute a given task. These types of information can be used to change the behaviour of an application running on a mobile device in order to adapt its use to different situations.

Context-awareness is also crucial in mobile service provisioning and orchestration (Kouadri Mostéfaoui, 2003). Services providers and service composition platforms can take advantage of additional information to provide flexible and adaptive services (Baresi et al., 2003). There are many ways of capturing and representing context as well as techniques to reason about context. This topic is of course important but outside of the scope of this paper. The interested reader might refer to (Baldauf and Dustdar, 2004).

An important dimension of context is the user's actual spatio-temporal information, which includes time-stamped geographical coordinates of the user's location and motion parameters (e.g. speed, acceleration, direction). This type of information plays an essential role in mobile ubiquitous computing applications as it provides the user not only with a context but also with an additional input modality, which can be used alone or in combination with other ordinary, possibly unobtrusive and natural input modalities (e.g. voice, gesture recognition). In the simplest cases, users can simply provide as an input their current geographical location obtained either from an outdoor localization system (e.g. GPS or Wireless Cell Triangulation) or from physical presence/motion detection system in indoor situations (e.g. "smart-buildings" equipped with RFID readers). Mobile ubiquitous applications usually exploit geographical information in order to allow the user access geo-contextualized services, publish multimedia data captured during their roaming (e.g. geo-tagging and geo-blogging) and interact with remote applications by motion.

In this paper we describe the basic features of the RoamBlog architecture on top of which a wide class of mobile ubiquitous computing applications can be designed and implemented. The types of applications we are targeting are those requiring full user interactions with a number of integrated geographically contextualized services. Interaction can be achieved either by using ordinary user interfaces on mobile computing devices or by the recognition of user's goals and intentions from physical motion by means of what we call the Kinetic User Interface (KUI). Three motivating usage scenarios are also described which demand the particular features of the architecture we proposing.

1.1 Motivations and Goals

Our involvement in the research area of "Mobile and Ubiquitous Services and Applications" is motivated by the increasing availability of wireless Internet connectivity (e.g. GPRS/UTMS, WiFi, EDGE) and by the upcoming availability of enhanced satellite navigation systems, such as Galileo[1]. The success of Internet "blogs" and the increasing availability of Internet-enabled mobile phones suggest us to consider types of applicative scenarios where mobile Internet

[1]http://www.esa.int/esaNA/galileo.html

users can not only access to contextualized content and services but also produce them while roaming, without taking care of all details of the process (i.e. manually establishing an internet connection, authenticating, typing text, and uploading media). In fact, RoamBlog allows the continuous tracking of mobile internet users and the transfer multimedia data enriched with timestamps and geographical coordinates onto a remote server which might hosts, for instance, the users' blog pages.

The RoamBlog architecture thus enables scenarios in which the easy authoring of spatio-temporally annotated web content is made possible. For instance, combining RoamBlog applications with "mashups" techniques[2] on blog clients, readers can enjoy a really immersive experience by suitably retrieve the author's data scattered over time and places, and replay the author's roaming experience. Moreover, if real-time geographical tracking is enabled, the blog's readers can follow the authors and interact with them during their roaming directly from a web/desktop/mobile interface, either by following their paths on a three dimensional map (e.g. Google Earth[3]) containing the geo-located multimedia data, or by suggesting directions and targets which are delivered to the roaming users in a suitable way (e.g. SMS, WAP, Instant Messaging, Voice over IP, ordinary phone calls).

Another type of scenario is geo-contextualized service provisioning. Services delivered on mobile devices can be either automatically invoked or geo-contextualized when explicitly requested. In the first case, when the user motion or location is detected, such as entering in a building or a room or passing by a given landmark motion is interpreted by the ubiquitous application as context shift. In such a case, subscribed context-aware services are automatically invoked and delivered. Additionally, the application can perform inferences on the context change history and recognize implicit user's goals and intentions. This type of scenario includes situations like guided tours, remote assistance in health institutions, smart home/building environments, etc.

Our goal is to elaborate the concept of Kinetic User Interface as well as a middleware architecture to support it. KUI is a user interface for ubiquitous computing applications where physical motion is used as a main input device. Motion is recognized by local or global positioning devices and location information is made available to pervasive applications that will react as in a usual mouse-based desktop interface by selecting geo-located items, showing pop-up information, and more generally triggering geo-contextualized actions.

---

[2] http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)

[3] www.earth.google.com

1.2 Related Work

Outdoor and indoor motion tracking is an essential aspect of our work. Many commercial GPS tracking applications are already on the market such as Trimble Outdoors[4] or ESRI ArcPad[5]. Unfortunately very few of them are integrated with Internet service provisioning or serve as user interfaces. An interesting example of commercial indoor tracking system is provided by Ubisense[6] which proposes both a hardware and software solution. Ubisense allows users to wear a light smart-badge with high-accuracy indoor motion tracking. Ubisense provides APIs and visual tools for developing customized applications.

From application side, with the advent of Web2.0[7], a great deal of applied research and development has been recently devoted to embed Internet technology in everyday life, ranging from pure entertainment to critical applications such as healthcare, national security, military (Cáceres et al., 2006). The rationale behind these efforts is to provide the mobile internet user with a great flexibility in authoring, publishing and retrieving information, as well as accessing services that are relevant in a given situation. A remarkable example Web2.0 mobile application for geo-tagging is SocialLight[8], which allows the tagging of geographical location with multimedia tags (i.e. shadow-tags). The roaming users can geo-tag a place either when they are physically present using a mobile phone, or by attaching the tag on the SocialLight web page by a GoogleMap mash-up. Compared to RoamBlog, SocialLight has similar design goals, but only fulfil them partially. Therefore, we are aimed at a more general framework as the one recently proposed for Blogjects (Bleecker, 2006), which includes SocialLight-like applications as special cases.

Although context-aware mobile (or nomadic) computing is still in its infancy, a number of interesting ad-hoc applications has been developed in recent years, mostly in the academic research domain (see Chen and Kotz, 2000 for a survey). Academic research projects such as Mobile Media Metadata[9] (Davis et al., 2004) are among the first attempts in developing this new trend of pervasive mobile internet technology, while we expect to more similar projects to be developed in the next future relying on pervasive positioning technology (Schilit et al., 2003; Ashbrook et al., 2006). We observe, however, that the main focus in context-aware mobile computing is on adapting mobile applications to one particular context

---

[4] http://www.trimbleoutdoors.com/TrimbleOutdoors.aspx

[5] http://www.esri.com/software/arcgis/about/arcpad.html

[6] http://www.ubisense.net/

[7] http://en.wikipedia.org/wiki/Web_2.0

[8] http://www.socialight.com

[9] http://garage.sims.berkeley.edu/research.cfm#MMM

at the time rather than taking into account context shift and context history. Towards this alternative direction an interesting work is that of Cyberguide (Abowd et al., 1996), Cyberguide is one of the few attempts in taking user's motion into account (see also Schilit, 1995). A tourist equipped with indoor (IR beacons) and outdoor (GPS) localization devices can receive relevant information on a PDA and feed a trip journal.

2 The Kinetic User Interface

We observe nowadays a great effort in porting ordinary (i.e. PC-based) user interfaces to mobile computing devices (e.g. PDA, SmartPhones, Media Players). It is clear however that new types of interaction are required in order to unleash the usability of ubiquitous computing applications. Most of current context-aware mobile applications are based on small computer devices, but there is not much difference between a desktop PC and a PDA except the size and the responsiveness of the graphical interface. With the exception of sophisticated wearable interfaces equipped of multiple sensors and virtual reality rendering devices, input modalities for hand-held computers are limited to standard GUI with additional handwriting and voice recognition.

We believe that physical motion over the geographical dimension can itself be used as an additional input modality in context-aware mobile applications. We claim that geographical motion can be used not only as a source of contextual information but also as an input modality that reflects the user's goals and intentions, and we describe three scenarios where spatial motion plays an essential role. In other words, we would like to push the user to intentionally cause events by moving from one place to another, by following a certain path or by performing a given motion pattern.

We elaborate here the concept of "Kinetic User Interface" (KUI) as a way of endorsing the Ubiquitous Computing vision and the new vision of Continuous Computing (Roush, 2006). KUI is neither a Graphical User Interface (GUI) nor a Tangible User Interface (Holmquist et al., 2005), but rather an interface through which the *motion*, in terms of physical displacement within the environment, determines the execution of actions (e.g. service requests, database updates). In other words, motion is considered as an input for an application.

Similarly to "hovering" the mouse over a desktop, the user can either trigger events from the environment by just moving from a location to another, or just "passing by" a given object or landmark. The user can "click" the physical space by executing actions/operations on physical objects such as grabbing, touching, moving, juxtaposing, and operating portable devices through their own interfaces. Following a path or executing a pre-defined motion

pattern can be related to mouse “gestures” and consequently trigger reactions by the system. This way motion becomes a full-fledged interaction modality that can be afforded by the user, which can be used alone or in combination with other modalities.

Our approach builds on and extends the notion of context-awareness as defined in (Dey et al., 2001) where the focus is shifted from on-demand contextualized services only to include context-triggered ones. Context-aware applications traditionally make use of a static snapshot of context with the aim of contextualizing the service offered. In contrast, motion in a KUI represents the principal way of (embodied) interaction: context changes trigger system's reactions directly, rather than only providing a source of complementary information used to deliver a service adapted upon user's request. In particular we focus on an explicit and deliberate *change of context*; in other words, the user is “aware” what the current context is and decides to act in order to cause a reaction from the computational environment. For instance, a typical context change situation can be detected when a person is moving from an indoor to an outdoor environment.

To make a parallel with ordinary, mouse-controlled user interfaces, moving the pointer on the screen is an intentional act recognized by the system that can trigger reactions specified in the application currently running on the computer. Additionally, the system might be able to recognize and make use of several other parameters of motion like speed, acceleration, pauses, direction, etc. Although the user might be not aware of what reactions will be caused by its moving, he is indeed aware that motion will be taken into account by the system. Furthermore, a mouse-based user interface usually shows a predictable behaviour (i.e. hovering the mouse does not usually cause unexpected reactions from the system); being predictable is an essential feature of user interfaces, because it allows the user to feel progressively comfortable in using the device: a suitable amount of feedback is required to ease user's understanding of the system (i.e. the pointer arrow is displayed according to mouse motion).

In case of body motion in a physical space, feedback cannot be given in the same way as for pointing devices on computer displays. Generally, we need to give back only the minimal amount of information required to inform the user that his body motion has been recognized by the system, because the system should avoid interfering too much with user's current activity. Nevertheless, a feedback control mechanism is necessary for other reasons such as privacy: to grant a certain level of protection, the user must be notified somehow when his/her presence and motion is being tracked. Indeed, he/she must be always given the possibility to stop the tracking, more or less in the same way as the user could stop using the

mouse and keep only using the keyboard (which might in turn render the interaction with the computer very difficult if not impossible).

Motion alone is not sufficient for carrying out a complete interaction with an “ambient” computer, in a similar way as it might just be insufficient and unreliable to use a mouse without its buttons. Furthermore, relying only on motion can make the interface difficult to interact with, and misinterpretations of movements can lead to unexpected results. For this reason, the KUI should also support other interaction modes such as voice, tangible objects or gesture recognition in order to execute complete actions recognized by the system. For instance, it might be conceivable that in a Smart Home environment, the user asks the system to move a media being played in the room where is currently located to another room to which he/she is moving (provided that the media can be played with suitable devices in both rooms), by just carrying a meaningful object (e.g. the remote control).

We distinguish two main types of physical motion detection:

> Discrete motion detection: one single step of motion is detected no matter of what trajectory is followed between the start point and the endpoint. This is the case moving from a room to another or passing by a landmark.
>
> Continuous motion detection: the motion trajectory is reconstructed by the system by interpolating points got from periodical probes. This is the case, for instance, of a travelling vehicle tracked by GPS, or a moving person spotted on a video camera stream. Several parameters can be recorded such as speed, acceleration, direction, height, etc.

Services offered by the back-end application in reaction to events produced by the KUI can be divided in two classes:

> subscribed: the user has previously subscribed to a service and the service is activated when a change in the context is detected.
>
> on-demand: the user makes a request and the systems provides the service by taking into account the current context.

Subscribed services are normally delivered as reactions to notifications of context change (callbacks), whereas on-demand services only require the ability to access context information (current location, time, etc.).

As motion detection itself is not sufficient and sometimes impractical to interact with the application, an additional source of control can be provided by physical sensors (e.g.

phidgets[10]), or system-recognized "smart" objects. Coupling different input sources with body motion gives the user additional control over the system: in the previous Smart Home example, carrying the remote control is interpreted as "dragging" the media being shown from one room and "dropping" to another. Moving without the remote control should not trigger any media related "Drag&Drop" action. It is important to note that the "Drag&Drop" feature has to be programmed as part of application's KUI, which would then generate an event in response to the context change event (e.g. moving from one room to another with the remote control). The remote control object is what we call a "cursor" which is linked to objects that can be moved but cannot move by themselves.

The remote control example also shows how our notion of context can also describe the physical or logical relations between objects (e.g. containment, proximity, adjacency, ownership), and the fact that it is possible to conceive a nested or linked contexts views (e.g. nested containers). The fact that a person is moving from one room to another constitutes a context, if the person is also carrying an object can be considered as a particular sub-context of the former instead of an unrelated new context. Also, the co-presence in the same location of two or more object can be used to logically bind the objects together. For instance, if a person (a cursor object) and a remote control (a dragged object) are moving into the same location, the system can infer a link between them. In contrast, the link can be broken if the cursor object moves out without the dragged object. As pointed out before, carrying a detectable object while moving can lead to a system reaction (corresponding to the notion of "dragging" in the mouse-based GUI).

It is worth to note that KUI supports the "situated plan" view (Suchman, 1987). In fact, the application constantly monitors the user's behaviour (at least for what concerns the part of context which is submitted to the KUI-manager) and it does not follow a "rigid" plan. Rather it is "flexible" and "adaptive" in the sense that it might assume an explicit overall goal communicated by the user and it "collaborates" with the user to its achievement. Actions that are recognized as not being part of the workflow related to the stipulated goal are just ignored. Applications are responsible to select "relevant" actions. Specifically, actions are interpreted by the application and filtered according to contextual information (e.g. the action of stopping when there are no relevant nearby landmarks is filtered out).

3 RoamBlog Architecture

The RoamBlog architecture is a layered middleware for mobile ubiquitous computing applications based on standard enterprise application server architectures that implements

[10] http://www.phidgets.com/

the KUI interface. RoamBlog allows the geographical contextualization of mobile services (e.g. a service requested by a user in a given time and place) as well as the integration of geographical information into back-end applications (e.g. data-based updates triggered by motion detection). Mobile devices carried by the roaming user can be connected to the RoamBlog architecture as either client or servers depending on the above two situations. Corresponding to the two types of usage there are two mechanisms for exploiting and producing geographical information (coordinates):

- Pull: the coordinates are obtained from the proximity of an electronically tagged object (e.g. a badge) to a geo-referenced device (e.g. a RFID reader). The information read from the object is used to create a logical link between the user carrying the tagged object and the application that the user is currently using (provided that the reader is part of the application's infrastructure and the badge is bound to the user who is currently requesting the service). For instance, if a service is requested on a PDA just after that a RFID badge has been detected, the service will be aware of the current location of the PDA and the service will be contextualized accordingly.
- Push: the coordinates are generated by the user's device (which knows its own location) and sent to the application. In this case, the geographical information is locally gathered (e.g. a GPS receiver, fixed location RFID tag) and sent either as a parameter of service invocation or used in a KUI interaction.

The KUI interface is integrated in the RoamBlog three-layered architecture (as shown in Figure 1). The *Topology Layer* adds an overlay structure to the physical world made of references (geographical points) or aggregate references (areas): this information is linked with localization objects in the real world such as fixed RFID antennas, GPS receivers, etc.

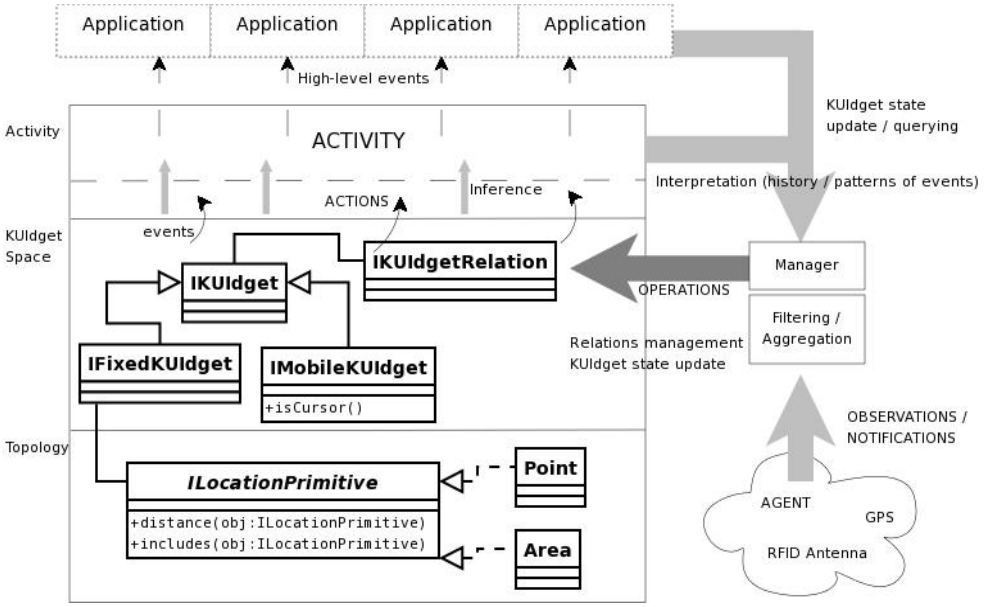

Figure 1. The three-layered RoamBlog architecture

The *KUIdget Space Layer* is an object-oriented framework that contains a set the context-carrying objects (KUIdgets) found in the real world that the service provider is interested in; a KUIdget can reflect a real entity (i.e. persons, places, and physical objects) or be completely virtual (e.g. a geo-tagged media). Each KUIdget is identified with a *universally unique identifiers* (UUID), and can be linked with a data structure in the Topology Layer to provide direct localization; indirect localization is obtained from other KUIdgets through chains of KUI relations (e.g. containment or proximity): for example, to find the location of the mobile phone of a user driving in a GPS-equipped car, it is possible to inherit the position from the container cursor KUIdgets (in this case the user from the car, and the mobile phone from the user). We also propose the use of weights so that the relationship and dependence between objects can be better managed and exploited: high values means that there is a very strong location-relationship between two objects, whereas with low values it is possible to express a weaker relationship (objects are loosely coupled), meaning that indirect positioning cannot be inferred reliably. Changes in relations and state (represented as object properties) of KUIdget generate events that can be captured by related KUIdgets or by the upper layer, the *Activity layer*, which manages higher-level semantic contexts and emits context change signals that are sent to the applications.

Relations can be created either by the KUIdget's internal logic in response to events (or by fetching information from the lower level) or by upper layers (to reflect an inferred situation). For example, in the first case a *containment* relation is created when a use carrying a RFID badge enters a room equipped with a RFID antenna at the door. In the second case the user explicitly communicate his/her position to the application (for instance, using a mobile

communication device), and the system can relate her with KUIdgets in the range. The importance of keeping relations in the KUIdget Space Layer derives from the fact that when context changes within a container KUIdget are projected onto all the contained KUIdgets which, in turn, might themselves generate a context change event. This is important because an application can be only aware of the contained KUIdget's context dynamics and not of the containing one, which is nonetheless responsible of providing context information to the inner one.

RoamBlog also enables social interaction. When two users both carrying a KUI-enabled device find themselves close to each other or in the same place (e.g. the same room, the same train car, or attending the same event), if they both subscribed to the same RoamBlog-based social network service they can be alerted of their co-presence. Depending on the subscribed service they can be prompted with relevant information and possibly get to know each other. The user might, for instance, specify filtering criteria, temporarily disable the subscription (e.g. do not disturb), or even request a server to locate other users who match certain criteria (e.g. find any Italian speaking users subscribing a specific service within a range of 1 Km.).

4 Pilot Projects

We present in this section two projects that demonstrate the use of KUI and contextualized service provisioning. While they both exploit the concepts outlined in the RoamBlog architecture, they only illustrate them separately. In particular, the UbiBadge project adopts the indoor discrete motion KUI interface with contextual service provisioning, while GlideTrack illustrates a case for the outdoor continuous KUI.

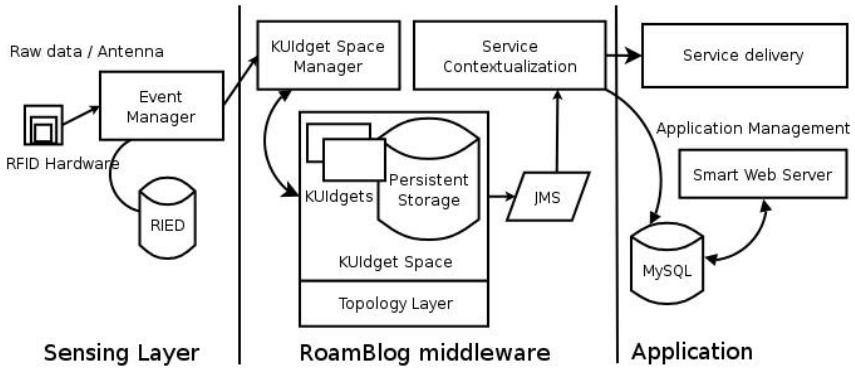

Figure 2. UniBadge architecture

4.1 UbiBadge

We outline the main aspect of a pilot project conducted at the University of Fribourg and based on the RoamBlog architecture. The idea behind this project is to offer contextualized services to students carrying an RFID badge, based on their wandering in the University's building. Events can be triggered, for example, by stopping at the student association billboard, entering a room or by leaving it. These events cause a series of contextualized reactions, for example the delivery of the today's menu on the student's mobile phone when entering the university's canteen, or sending news when the student stops in front of a notice board. The UniBadge architecture sits on top of the RoamBlog middleware, where the University's topology (rooms, landmarks) and mobile entities (students) are abstracted to KUIdgets. The localization service is provided by the RFID Locator detailed in (Fuhrer et al., 2006) and also developed at the University of Fribourg. Rough events gathered from RFID hardware are sent to corresponding KUIdgets which then spawn higher-level events (e.g. entering or leaving a room) that are captured by the Activity Layer. Based on this information, the latter updates the actual context and informs the application of the context changes, in order to deliver the corresponding service.

4.2 GlideTrack

The GlideTrack project, whose architecture is shown in Figure 3, is an example of the use of KUI-based interaction for the case of continuous motion. The project is meant to provide a simple and portable solution for people to get traced during a trip. The users of the GlideTrack application have an associated KUIdget whose context can be updated by simply receiving the actual coordinates from a GPS enabled cell phone through a GPRS internet connection. Context changes are stored in a database, and can be retrieved and mashed up onto a geographical map (e.g. Google Earth).

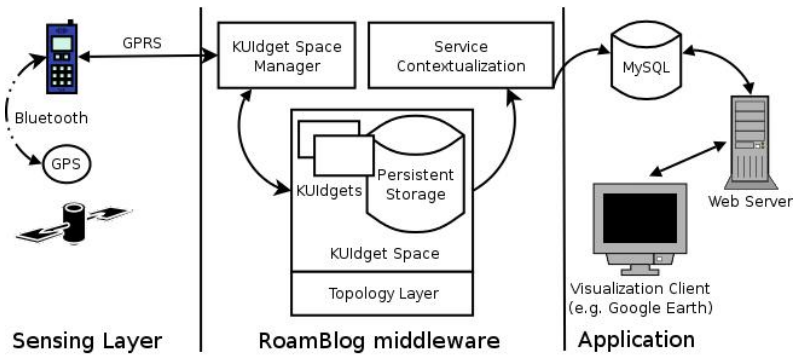

Figure 3. GlideTrack architecture

5 UbiTour Scenario

We present an overview of an ongoing project on e-Tourism, combining indoor-outdoor KUI, geo-blogging and geo-contextualized service provisioning.

5.1 Scenario description

Discovery and surprise are among the main motivations of tourists. Tourists like to see new things and find themselves in unexpected, but still controlled situations. On the one hand, if tourists had to fully plan their vacations then they are likely to loose the surprise dimension and will find their vacation less satisfying as expected. On the other hand, tourists would like to have their trips organized by a tour operator in order to minimize the risk of finding themselves in unexpected bad situations. As an optimal trade off between fully planned and completely blind vacations, tourists might like to have a certain degree of flexibility, and thus have the possibility to cancel some already planned travel decisions that might prevent them to enjoy unexpected situations which they could not be aware of until the actual time of the trip. Finding the right trade-off between a rigid, full organized tour, and a fully flexible, unorganized vacation, is the problem we would like to address here. UbiTour is our solution to this problem. UbiTour allows its users to start with a minimally organized vacation and incrementally and contextually plan the next steps “on-the-go”. Moreover, UbiTour provides contextual help and recommendations which the user is still free to adopt or ignore.

This is clearly a scenario where RoamBlog could play a fundamental role. To some extent, the user is always (transparently and unobtrusively) followed by a virtual assistant which can be either contextually queried or can take initiative in reaction to relevant context changes. Moreover, UbiTour tracks users’ displacements and collects data about their experiences by updating travel blogs. In this way, users do not have to worry about collecting their travel notes in a journal or upload their pictures/movies on their blogs[11]. As a desired side effect, safety issues are also addressed in this scenario. The interruption of the information flow from the roaming user might signal an emergency situation that could be detected and pro-actively handled by the UbiTour assistant. The UbiTour application defines a geographical KUI space where the user can freely move, select items, receive and send information using personal devices, and interact with local connected devices (e.g. touch screens).

As an example of the types of services offered by UbiTour, imagine that a user spent a whole day driving along the Californian coast where she stopped several times as she spotted an interesting landscape. At 8PM she is tired and wants to find an overnight accommodation in

[11] Nowadays an important concern of tourists is allowing relatives and friends to remotely follow them during their vacation. This is typically done by periodical calls or manually updating a travel journal in a public Blog.

the closest urban center. She is approaching San Luis Obispo and she informs the system that she needs a place to sleep. The system already knows her accommodation preferences and when she passes by a hotel she spots (by decreasing the speed or stopping nearby) the system checks in a database if the hotel suits her preferences and if a room is available. It then informs her of the research results and starts a confirmation dialogue using a suitable modality (e.g. voice).

Let's look now the details what happens at RoamBlog level. First, the user activates the KUI interface by informing the UbiTour application that she is looking for a Motel where she intends to spend the night. UbiTour selects a number of relevant landmarks (i.e. Motels connected to the Ubitour network) and start recognizing when the user is approaching or stopping at one of them by listening to location change events generated by the KUI middleware. Technically, the KUI-space Layer sends events when the user enters or leaves the area associated with landmark (i.e. a proximity relation has been created). These events are mapped onto higher-level actions of the "Motel selection" task in the overall activity of "finding an overnight accommodation". The next steps would be making the reservation and checking in by possibly using an ordinary interface on a mobile device.

6 Conclusions

The original contribution of this work to geo-contextualized service provisioning is the seamless and transparent integration of indoor and outdoor motion detection within context-aware mobile Internet infrastructure. The RoamBlog architecture enables users of mobile Internet applications to interact with a back-end system by means of a new user interface based on motion tracking, the Kinetic User Interface, with no manual switching between indoor and outdoor modality. This brings the advantage of reduced user intervention on the carried mobile devices, as well as the better awareness of the services being activated by the user's motion. We described the basic features of the RoamBlog architecture as well as three scenarios where this type of architecture shows its practical usefulness.

6.1 Future Works

RoamBlog is a new project and thus full of shortcomings and possible extensions. It does not make sense to fully address them here. However, it is worth to note that RoamBlog stems from ideas related to Activity Theory (Bodker, 1991; Nardi, 1996) and Multi-Agent Systems (Ricci et al., 2006). Our goal is to better ground the RoamBlog framework on insightful theories and develop a clear software engineering methodology for the rational design of ubiquitous computing applications. We are also interested in usability issue as, for instance,

in studying the new types of affordances[12] and interfaces that this kind of "invisible computer" might offer. As pointed out by (Norman, 1999), the power of ubiquitous computing lies in the power of the infrastructure. We add that the infrastructure as it is now is not "usable" in the sense that users are still constrained by the old-fashioned WIMP[13] user interfaces on mobile devices. In constrast, Information Appliances (i.e. intelligent and networked consumer appliances) and Blogjects (i.e. location-aware real or virtual active object with Internet connectivity) will be the types of devices we will have to cope with. The productive coordination among these entities in is not trivial and will definitely lead to the creation new concepts in distributed computing and human computer interaction.

6.2 Acknowledgements

RoamBlog is part of the COPE project on *Computing in the Physical Environment: Context-Aware and Intelligent Acting*, currently on-going at the Pervasive & Artificial Intelligence research group[14], Department of Computer Science, University of Fribourg, directed by Prof. Béat Hirsbrunner.

References

G. D. Abowd, C.G. Atkeson, J.I. Hong, S. Long, R. Kooper, M. Pinkerton: Cyberguide: A mobile context-aware tour guide. Wireless Networks 3(5): 421-433 (1997)

D. Ashbrook, K. Lyons and J. Clawson. 2006. Capturing Experiences Anytime, Anywhere. In IEEE Pervasive Computing (Vol. 5, No. 2) April-June 2006. pp. 8-11.

M. Baldauf, S. Dustdar, A Survey on Context-Aware Systems. Technical Report n. TUV-1841-2004-24 (30 November 2004), University of Wien.

L. Baresi, D. Bianchini, V. De Antonellis, M.G. Fugini, B. Pernici, and P. Plebani, Context-aware Composition of E-Services, in Proceedings of TES03 (Technologies on e-Services) Workshop in conjunction with VLDB 2003, Berlin, Germany, September 2003, LNCS 2819, pp. 28-41, Springer-Verlag, 2003.

J. Bleecker, Why Things Matter: A Manifesto for Networked Objects — Cohabiting with Pigeons, Arphids and Aibos in the Internet of Things, 2006.
http://research.techkwondo.com/files/WhyThingsMatter.pdf.

S. Bodker, Through the Interface. A Human Activity Approach to User Interface Design. Lawrence Erlbaum Associates, 1991.

C. Cáceres, A. Fernández, S. Ossowski, CASCOM - Context-Aware Health-Care Service Co-ordination in Mobile Computing Environments", (URJC). ERCIM News num. 60, 2006.

G. Chen and D. Kotz. 2000. A Survey of Contex-Aware Mobile Computing Research. Darthmouth Science Technical Report TR2000-381.

[12] Affordances are operations made available to users by an object which are self-evident by its essential features (Gibson, 1979).
[13] WIMP stands for Windows, Icons, Menus, and Pointing.
[14] http://diuf.unifr.ch/pai/research/

M. Davis et al. 2004. From Context to Content: Leveraging Context to Infer Media Metadata. In Proceedings of the 12th Annual ACM International Conference on Multimedia (MM 2004), ACM Press, 2004, pp. 188–195.

A.K. Dey, D. Salber, G.D. Abowd, A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications, anchor article of a special issue on Context-Aware Computing of Human-Computer Interaction (HCI) Journal, Vol. 16 (2-4), 2001, pp. 97-166.

P. Fuhrer, D. Guinard, and O. Liechti. 2006. RFID: From Concepts to Concrete Implementation. In Proceedings of the International Conference on Advances in the Internet, Processing, Systems and Interdisciplinary Research, IPSI - 2006 Marbella, February 10-13, 2006.

J.J. Gibson. The Ecological Approach to Visual Perception, Lawrence Eribaum, Hillsdale, NJ. 1979.

L. Holmquist, A. Schmidt, B. Ullmer. Tangible interfaces in perspective: Guest editors' introduction. Personal and Ubiquitous Computing 8(5), 2004. pp. 291-293.

S. Kouadri Mostéfaoui, "Towards a Context-Oriented Services Discovery and Composition Framework," Proc. AI Moves to IA: Workshop Artificial Intelligence, Information Access, and Mobile Computing, held in conjunction with the 18th Int'l Joint Conf. Artificial Intelligence (IJCAI '03), 2003.

B. Nardi, Context and Consciousness: Activity Theory and Human-Computer Interaction. Cambridge, MA. MIT Press, 1996.

D.A. Norman, The Invisible Computer. Cambridge, MA. MIT Press, 1999.

P. Prekop, and M. Brumett. Activities, Context and Ubiquitous Computing. Computer Communications, special Issue on Ubiquitous Computing. 2002.

A. Ricci, M. Viroli, A. Omicini. Construenda est CArtAgO: Toward an Infrastructure for Artifacts in MAS. In Proceedings of European Meeting on Cybernetics and Systems Research (EMCSR'06), April 18 - 21, 2006. University of Wien.

W. Roush, What is Continuous Computing? In Continuous Computing Blog, http://www.continuousblog.net/2005/05/what_is_continu.html

B.N. Schilit, A. LaMarca, G. Borriello, W.G. Griswold, D. McDonald, E. McDonald, A. Balachandran, J. Hong, V. Iverson. 2003. Challenge: Ubiquitous Location-Aware Computing and the "Place Lab" Initiative. In Proceedings of WMASH'03, September 19, 2003, San Diego, California, USA.

B.N. Schilit, System architecture for context-aware mobile computing. Ph.D. thesis, Columbia University, 1995. http://citeseer.ist.psu.edu/schilit95system.html.

L. Suchman, Plans and Situated Action: The problem of human-machine communication. Cambridge, UK: Cambridge Univ. Press, 1987.

M. Weiser, The Computer for the 21st Century. Scientific American 265, No. 3, 94-104. September 1991.

M. Weiser, Hot topic: Ubiquitous computing. IEEE Computer, pages 71--72, October 1993.