Kinetic User Interface: Interaction through Motion for Pervasive Computing Systems

Pascal Bruegger and Béat Hirsbrunner

Pervasive and Intelligence Research Group
Department of Informatics - University of Fribourg - Switzerland
{pascal.bruegger,beat.hirsbrunner}@unifr.ch

Abstract. We present in this paper a semantic model for the conception of pervasive computing systems based on object or user's motions. We describe a system made of moving entities, observers and views. More specifically, we focus on the tracking of implicit interaction between entities and their environment. We integrate the user's motion as primary input modality as well as the contexts in which the interaction takes place. We have combined the user activities with contexts to create situations. We illustrate this new concept of motion-awareness with examples of applications built on this model.

Keywords: Pervasive computing, Ubiquitous computing, Motion-awareness, Kinetic User Interface, HCI.

1 Introduction

In this paper, we explore a new human-computer interaction (HCI) paradigm for pervasive computing systems where *location-awareness* and *motion tracking* are considered as *first input modality*. We call it Kinetic User Interface (KUI) [1].

Nowadays many projects such as EasyLiving [2] or GUIDE [3] have developed Ubicomp[1] technologies like mobile devices or applications and have enhanced human experience for instance by providing contextualised services mainly according to user's location. However, most of current context-aware systems are limited to external parameters and do not take into account user-centric dimensions. In our model, we consider user's activity as a way to reflect its goal and intention.

The paper formalizes KUI as a system composed of entities and observers. Kinetic objects (entities), possibly living things, interacting naturally with their environment are observed by agents (observers) which analyse their activities and contexts. The model focuses on *implicit* interaction and *unobtrusive interface* of motion-aware computing systems. The challenge consists in modelling the physical "world" in which entities live and interact into a conceptual system representing this world in a simple and flexible manner. We propose a generic model and a programming framework that can be used to develop motion-aware and situation-aware applications.

In section 2, we define our model of system made of on entities, observers and views. In section 3, we present the concept of motion-awareness. In section 4, we

[1] Ubiquitous Computing equivalent to what we have called in this paper pervasive computing.

C. Stephanidis (Ed.): Universal Access in HCI, Part II, HCII 2009, LNCS 5615, pp. 297–306, 2009.
© Springer-Verlag Berlin Heidelberg 2009

describe the implementation of this model and in section 5, we present the application domain with the description of three KUI enabled projects based on our model.

2 KUI Model: A Systemic Approach

The approach we have chosen for our semantic model is based on the *General System Theory* (GST) and the research of three authors [4],[5],[6] who propose interesting visions of systems.

For Alain Bouvier ([6], p.18), a system (complex organised unit) is a set of **elements** in **dynamic interaction**, organised to reach a certain goal and differentiated within its environment. It has an identity and represents a "finalised whole". General System Theory (GST) defined by von Bertalanffy [4] describes systems in sciences such as biology, chemistry, physics, psychology. GST gives the framework and the concepts to model specific systems studied in sciences. There exist different types of systems such as **inert** or dead, **living** or evolutionary, **open** (exchanging matter with its environment) or **close**. For instance, we can see the world as a whole extremely complex living but close system. For physicians, this perception of the world is not correct and this vision is reductive. The world (our planet) is one component of the solar system and it is part of the equilibrium of this system. Boulding in [5] writes: *an "individual" - atom, molecule, animal, man, crystal - (entity) interact with its environment in almost all disciplines*. Each of these individuals exhibits "behaviour", action or change and this behaviour is considered to be related in some way to the environment of the individual, that is, with other individuals with which it comes into contact or into some relationship. The important points in Boulding's definition is that:

- The entity's actions (activities, behaviour) are related to their environment;
- Entities come into relationship.

In KUI, systems are open and dynamic (living). Their complexity evolves over time with respect to their components. Components can join and leave systems, increasing or reducing their size. We have included two concepts which are not present in the chosen author's definitions:

- The observer: who/what is observing the system;
- The view: observer 's point of view.

We define a system as a set of observable, interacting and interdependent objects, physical or virtual, forming an integrated whole. The system includes different types of objects: entities, observers, and views.

2.1 Entities

Entities are the observable elements of the system. They can be physical or virtual, living things (humans, animals), moving objects (cars, planes), places (rooms, buildings). An entity is made of **contexts** and do **activities** (Fig. 1).

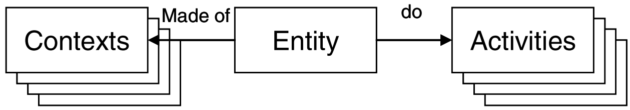

Fig. 1. Basic structure of an entity

Contexts. As defined by A. Dey et al. [8], a context is any information that can be used to characterise the situation of an entity. In our model, the contexts are used to define the attributes of an entity. Contexts do not include the activity. The activity is influenced by the environment and therefore by the contexts in which it is made. We will see later that contexts provide relevant information to the observer in the situation analysis. We use the following contexts in our model: Identity, Location, Role, Status, Structure, Relations.

Identity and location. The identity is the name of the entity and must be unique in order to be differentiated within the system [6]. The location is the address where the entity can be observed. The location is made of an address and a time. The dynamic behaviour of an entity makes the address possibly dynamic and time must be taken into consideration.

Role. We have defined two roles of entity in our model: 1) actor and 2) place. They indicate to the observer what it should focus on. For instance, motions and activities are the focus when the entity is an actor. Roles are dynamics and the entity, according to the point of view of the observer, is sometimes an actor and sometimes a place. For example, a cruise boat can be observed as an actor when we consider its kinetic properties (cruising around the globe) and it is considered as a place when we focus on its structure (passenger, cabins, decks).

Status. It provides entity kinetic information to the observer. An entity has two possible status: *mobile* (motion capabilities) or *static* (fixed).

Structure. The structure of an entity can be simple or complex. A simple entity does not contain other entities. It is called an *atom* (e.g. a walking person). At the opposite an entity with a complex structure is said *composed* (e.g. a house). A composed entity contains other entities and must have at least one atom. The structure of an entity is dynamic and evolves over time. For example, a house is made of rooms and contains inhabitants. Each time a person changes room, the structure is modified.

Relations. They determine the type of interaction the entity has with its environment. Relations provide information about the state of entities and contribute to evaluate the situation. We consider two types of relations between entities: *spatio-temporal* relations and *interactional* relations.

A spatio-temporal relation defines the "physical" connection between entities. When an actor is nearby a place or an other actor at the same time, a temporary relation exists. Our model of spatio-temporal relation is inspired from the spatial relationship used in GIS [10]. We differentiate 4 types of spatio-temporal relations:

1. Proximity (*Next To*)
2. Containment (*Inside*).
3. Contiguity (*Juxtaposed*)
4. Coincidence (*overlap*)

The relations 1,2 are created between an actor and an other actor or a place and the relations 3,4 concern places.

We call interactional any relation between actors needed to carry out complex activities. These relations are parameters used to determine the feasibility of an activity. We have identified 3 types of relations:

1. Collaborative
2. Causal
3. Conditional.

The collaborative relations are set to carry out activities that cannot be achieved by one actor. For instance, the activity of fixing a vertical pillar on the street requires the intervention of a crane driver who lifts up the pillar and maintains it stable while a worker is bolting the pillar on the plinth. We see in this case that the completion of the main activity is possible only if at least two specialised activities are combined.

A causal relation is present when the activity *A* causes the activity *B*. For instance, if a car (entity A) moves it implies that the driver (entity B) moves as well. Causal relation is useful to check the "validity" of the detected activity.

Conditional relations are created when activities have to be made in a given order. Like in causal relation, it allows to check the validity of an activity. Activity *B* can be done only if activity *A* is done before.

Activities in Places. Activities are controlled within a place. Places have rules that determine the authorised activities, the forbidden ones and the negotiated ones.

We introduce the concept of white and black activity lists. White-listed activities are the authorised activities that can be carried out with no reaction from the observer. They are accepted as such. Black-listed activities are, in contrary, forbidden and provoke an immediate reaction from the observer. We also take into consideration what we call the "grey" list. If an activity is not explicitly declared in the white or black lists then it is "negotiable" and gives the freedom to evaluate it and to infer on the situation. Activity lists (black and white) allow the observer to quickly react when something is going on in a place. The observer simply checks if the ongoing activity is explicitly present in one of the lists.

2.2 Observers and Views

In the previous section, we have detailed the system by its entities. The second part of the system consists of the entities observation.

Observers are the agents which observe the moving entities populating a system. They collect and analyse information (activities and contexts) of actors and places and possibly react on dangerous or inappropriate situations in which actors could be. Observers are specific and analyse one or a small number of situations. Our vision is to have more observers but less programming complexity per observer.

To illustrate this concept, we take the example of UN[2] observers placed at the border between two countries during a cease fire. Their role is to watch, to observe movements of troupes of both countries (the situation) and to react or notify when they detect violations of rules. These rules are established in advance and must be respected by the actors on the field. For instance soldiers must not cross the no-man's land. An UN observer analyses the actor's activities and contexts (location, time) and reports any detected incident or violation to the higher level, his hierarchical superior.

None intrusive behaviour of observers. Weiser in [9] has introduced the concept of calm technology. In his concept, the user is more and more surrounded by computing devices and sensors. It becomes a necessity to limit the direct interaction with the computing systems in order to avoid an unneeded cognitive load and let the user concentrate on its main activity. Our concept of observer is inspired from the Weiser's idea. There is no interference with actors and places: the observer reports only problematic situations to the higher level and let the application decide what to do.

Views. The entities are observed under certain points of view. Observers can select different points of view to analyse the same situation. Each point of view represents a *focus* on the situation. Many observers can use similar views for different situations analysis. A view is a *multi-dimensional* filter placed between an observer and the entities. It allows or constraints the observer to focus on a certain part of the system. The focus goes from the root structure, to one atom. We have 2 dimensions in our model of view: range and level.

The **range** is a parameter that influences the scope of the observation (e.g. the ocean or only a cruise boat) and the **level** is the parameter which gives the granularity of the observation (e.g. decks or decks and cabins or passengers).

For instance, a photographer uses different types of lenses according to the level of observation. The wide angle lens gives a large view of the landscape (the range) but looses details like bees on flowers. Now if the focus is a bee on a flower then a macro lens is needed. The level changes. Actually, the photographer cannot have at the same time the level of a bee and a wide landscape. This limitation range/level is solved in our model.

3 A Motion-Based Model for Situation Awareness

In this section, we start to define how the kinetic information of the different entities are processed and how from a simple motion a situation is derived. Context-aware system often consider the user's external parameters such as location, time, social information and activity to characterise a situation. In our model, we bring a new point of view in situation characterisation by separating the activity from contexts. Indeed, we consider that user's activity should be interpreted in their contexts in order to fully understand their situation.

Like the figure 2 shows, the motion-aware model is divided in 2 levels. At the entity level, we have the activities and contexts. It includes the motion detection. Situations are analysed at the observer level and are high-level semantic information.

---

[2] United Nations.

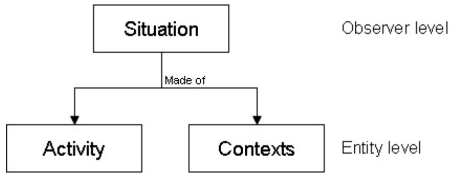

Fig. 2. Activity and contexts are component of a situation

Our situation-aware model is inspired from the Activity Theory presented by B. Nardi and K. Kuutti in [11],[12], Y. Li and J. Landay in [13] as well as the Situation Theory of J. Barwise et al. [14] and the work of S. Loke [15].

Situations. In [13], Y. Li and J. Landay propose a new interaction paradigm for Ubicomp based on activity (activity-based ubiquitous computing). In their model, the relation between activity and situation is defined as follow: *An activity evolves every time it is carried out in a particular situation.* A situation is a set of actions or tasks performed under certain circumstances. Circumstances are what we call contexts.

According to Loke, the notion of context is linked to the notion of situation [15]. He proposes the aggregation of contexts (perhaps varieties of) in order to determine the situation of entities. In that sense the situation is thought as being at an higher level than context. Loke makes a difference between activity and situation and considers an activity as a type of contextual information to characterise a situation.

Our model of situation combines the two visions (contexts and activities, Fig 3b) and we define it as follows: *A situation is any activity performed in contexts.*

Context-Awareness. The user is often unaware of the surrounding computing systems and do not feel the interaction with it. As mentioned by A. Dix [16], the main challenge of the pervasive computing is the *Where* computers are, the context-aware computing challenges are *What it means to interact* with computers. Context-aware applications uses contextual information to automatically do the right thing at the right time for the user [15]. Dey et al. [8] defines contexts as “Any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves […]”. They consider that the most important contexts are the location, identity, activity and time. This definition brings a new fundamental dimension in our model: the activity.

Activity. In [15], activity typically refers to actions or operations (Fig. 3b.) undertaken by human beings such as “cooking”, “running”, “reading”. For Yang Li and James Landay [13], activity like "running" is considered as an action focused on attaining an immediate goal. They consider, like Kuutti in [12], that an activity is the long-term transformation process of an object (e.g. a user’s body) oriented toward a motive (e.g. keeping fit). The notions of "long term" and "immediate" allow the separation of activities and actions. This raises some questions not answered in this paper: what do we consider as long term and immediate? When an action becomes an activity and vice-et-versa? In our model, we consider that an activity is made of detected motions aggregated into operations and actions and is an input for observers.

4 KUI Development Framework: uMove v2

uMove v2 is the 2nd JAVA™ based implementation of the KUI concepts [1],[17]. It offers to programmers a standard platform to develop KUI enabled applications. The framework is separated into three layers in order to have a clear separation of concerns (Fig. 3a).

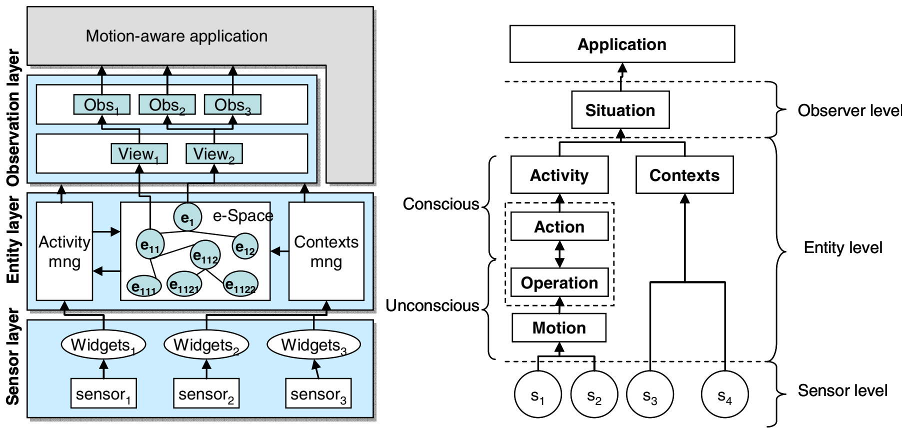

Fig. 3. a) uMove architecture, b) motion-aware model

The *sensor layer* contains all the widgets representing the logical abstraction of the sensors connected to the system. Then we have the *entity layer* in which we find the logical representation of the physical users or objects being observed. The activity manager aggregates the motion events into activities and makes them available to the observer. The contexts manager gets the sensor information, updates the entities and sends the information to the *observation layer* which analyses the current situation of the entity. Observers send events to the application according to the detected situations. This model allows the programmer to concentrate on the specific needs of the application without worrying about the communication between sensors (widgets), user or objects (entities) and their management (creation, remove, modification). The activity classes must be specifically developed and can be combined to enable complex motion pattern recognition. Observer and view classes, like activity classes, are developed for specific situations analysis.

5 Application Domain

Our KUI model focuses on applications which integrate motions in a large sense as main input modality. In the ubiGlide project [17], uMove allows the application to track hang-glider or paraglide motions. Based on their contexts the application infers and informs the pilot (Fig. 4) about potentially dangerous situations like flying over a no-fly zone, a national park or nearby a storm. In ubiShop project [18], the application is in charge to control the inventory of the fridge in a family house or a shared flat and to request one or more house/flat inhabitant to get missing items (milk, juice, eggs) according to their current location and activity. For instance the father quietly returning by foot from work and passing nearby a grocery shop will be informed by the system that items are needed. However the system does not react if it detects that the father is running or walking fast; it looks for somebody else. In Smart Heating System (SHS) project [19], the application must adapt the room temperature according to the profile of users in the room and their current activity. The application regulates the thermostatic valves of radiators keeping a comfortable temperature in the room and avoids the waste of energy.

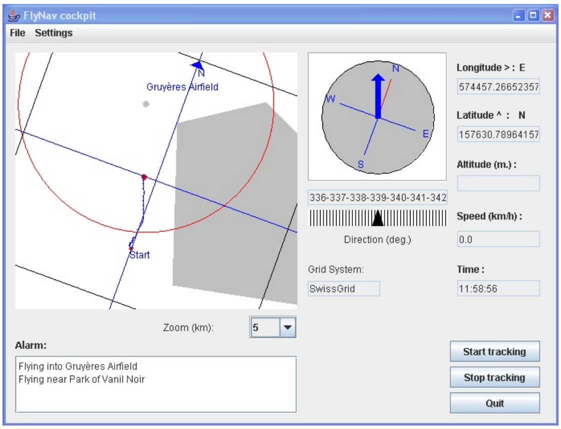

Fig. 4. ubiGlide - pilot's graphical user interface of FlyNav application

These projects validate uMove in three classes of applications. ubiGlide proposes a model of application for outdoor activity tracking such as trekking, mountain biking, sailing. These applications can prevent accidents by observing the user's behaviour in specific environments and by informing him/her about potential dangers. ubiShop is a validation scenario for applications that need to track motions and locations of entities in urban environments and distribute tasks in an opportunistic manner like for courier

Table 1. Overview of the three projects developed with uMove

Projects:	**ubiGlide[17]**	**ubiShop[18]**	**SHS[19]**
Entities	Flying objects, zones, mobile zones	People, zones, shops	People, rooms
Activities	Flying	Running, walking, standing	Quiet, active, slepping
Contexts	Flying objects, zones, mobile zones, location, speed	Time, location, speed	Location, time
Sensors	GPS	GPS, RFID	RFID, accelero-meter
Architecture	Distributed	Centralised, web based	Centralised
Environment	Outdoor	Indoor, outdoor	Indoor

or taxi services. Finally, SHS validates uMove in indoor environments and can be a model for applications providing service information on mobile devices or controlling the environment within a building according to user's location and activities. Table 1 shows the components of our model used or not in each of the three projects. Situation analysis is not yet implemented in these projects.

6 Conclusion

This paper has presented a new human-computer interaction paradigm where location-awareness and motion tracking are considered as first input modality. We call it Kinetic User Interface (KUI). We have presented the semantic model of KUI based on a systemic approach. uMove programming framework implementing KUI has been described and three projects using the KUI concepts and uMove have been presented. We believe that KUI concept and implementation can offer a good tool for developers to rapidly prototype applications that integrate motions of users and mobile objects as main implicit input modality.

Based on this new semantic model, uMove v2 has been finalised and as future work the three projects presented in section 4 will be upgraded to the new version. This will include the concepts of observers and views. Two other important challenges are planned in a near future: activity and situation modelling and implementation. In Smart Heating System, only 3 types of activities are taken into consideration and we will propose standard interaction patterns that can be used by developers in specific applications and extended types of activity recognition. We will provide guidelines to help programmers to properly define their system including the entities, observers and views before using uMove to implement the motion-aware application, activities tracking and situation analysis.

User studies must be also conducted in order to verify the concept of unobtrusive interfaces in particular in user's activities that request a high level of attention like flying, driving or manipulating dangerous equipments.

Acknowledgments. We would like to thank Denis Lalanne and Daniel Ostojic for the feedback and advises on this paper. This work is supported by the Swiss National Found for Scientific Research Grant n. 116355.

References

1. Pallotta, V., Bruegger, P., Hirsbrunner, B.: Kinetic user interfaces: Physical embodied interaction with mobile pervasive computing systems. In: Kouadri-Mostefaoui, S., Maamar, Z., Giaglis, G. (eds.) Advances in Ubiquitous Computing: Future Paradigms and Directions, ch. 7. IGI Publishing (2008)
2. Brumitt, B., Meyers, B., Krumm, J., Kern, A., Shafer, S.: EasyLiving: Technologies for Intelligent Environments. In: Thomas, P., Gellersen, H.-W. (eds.) HUC 2000. LNCS, vol. 1927, pp. 12–29. Springer, Heidelberg (2000)
3. Cheverst, K., Davies, N., Mitchell, K., Friday, E.A.: Developing a Context-aware Electronic Tourist Guide: Some Issues and Experience. In: Proceedings of CHI 2000, Netherlands (2000)

4. von Bertalanffy, L.: General System Theory. Foundations, Development, applications. George Braziller (1969)
5. Boulding, K.: General systems theory. The Skeleton of Science 2(3), 197–208 (1956)
6. Bouvier, A.: Management et projet. Hachette, Paris (1994)
7. Vallgårda. A: A framework of place as a tool for designing location-based applications. Excercept of Master Thesis (2006)
8. Dey, A., Abowd, E.D., Salber, G.D.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human Computer Interaction Journal 16, 97–166 (2001)
9. Weiser, M., Brown, J.S.: The coming age of calm technology, `http://www.cs.ucsb.edu/ebelding/courses/284/w04/papers/calm.pdf`
10. Calkins, H.W.: Entity-relationship modelling of spatial data for geographic information systems, `http://www.geo.unizh.ch/oai/spatialdb/ergis.pdf`
11. Nardi, B.A.: Context and Consciousness, vol. 1. MIT Press, Cambridge (1995)
12. Kuutti, K.: Activity Theory as a Potential Framework for Human-Computer Interaction Research. MIT Press, Cambridge (1996)
13. Li, Y., Landay, J.A.: Activity-based prototyping of ubicomp applications for long-lived, everyday human activities. In: CHI 2008: Proceeding of SIGCHI conference on Human factors in computing systems, New York, NY, USA, pp. 1303–1312 (2008)
14. Barwise, J., Gawron, J.M., Plotkin, G., Tutiya, S.: Situation Theory and its Applications. In: Center for the study of language and information - Stanford, vol. 2 (1991)
15. Loke, S.W.: Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective. The Knowledge Engineering Review, 213–233 (2004)
16. Dix, A., Finlay, J., Abowd, G.D., Beale, R.: Human-Computer Interaction. In: Pearson, 3rd edn. Prentice Hall, Englewood Cliffs (2004)
17. Bruegger, P., Pallotta, V., Hirsbrunner, B.: UbiGlide: a motion-aware personal flight assistant. In: Adjunct Proceedings of the 9th International Conference on Ubiquitous Computing, UBICOMP, Innsbruck, Austria, pp. 155–158 (2007)
18. della Bruna, D.: Ubiweb & Ubishop. Master project, Supervisors V. Pallotta, P. Bruegger, university of Fribourg – CH (2007)
19. Pallotta, V., Bruegger, P., Hirsbrunner, B.: Smart Heating Systems: optimizing heating systems by kinetic-awareness. In: 3rd IEEE International Conference on Digital Information Management (ICDIM 2008), pp. 887–892 (2008) ISBN: 978-1-4244-2917-2