Kinetic User Interfaces for Flexible Mobile Collaboration

Vincenzo Pallotta, Pascal Bruegger, Timothée Maret, Nicolas Martenet, Béat Hirsbrunner
Pervasive and Artificial Intelligence Research Group
Department of Computer Science
University of Fribourg, Switzerland
Vincenzo.Pallotta@unifr.ch

Abstract

In this paper we address the problem of collaborative workflow in mobile environments. Specifically, we present a solution for opportunistic task assignment to mobile users based on location, context and motion awareness. As an example of this approach, we present an application based on the new Kinetic User Interface framework where users participate to dynamic workflow through their displacement in designated active zones.

1. Introduction

A mobile organization could be described as an organization *"...where people, processes, technology and management support work (are) done anyplace/anytime*" [1]. Mobile organizations are supposed to be as effective as static teams when relaxing the constraint of co-presence (i.e., being at same place at the same time). Moreover, in certain activities the mobility is actually the key factor for obtaining the expected performance. Such activities include those in which tasks are dynamically assigned to individuals according to their current context (e.g. location, available resources and capabilities).

Workflow is commonly defined as a collection of tools that can be used to automate the sequence of actions or tasks used to run a formal process. It also includes tools for managing both the process itself and the coordination among actors, subtasks and resources.

In this paper we focus on *mobile workflow*, that is, the design of organization algorithms for the efficient workflow for *mobile organizations*. As pointed out in [2], from the ICT perspective, mobile organizations are characterized by a number of features (reported below):

- No fixed working space/time: The work can be done wherever and whenever is needed, according to given policies. These policies can be negotiated; they do not have to be rigid. For instance, home or public transportations use to be a workplace and, according to the type of the task assigned, the work can be executed on-demand or based on a personal schedule (meaning that the worker decides the work schedule).
- Internet-based processes: Processes are designed to be useful and accessible by both mobile workers and co-located workers, through the Internet infrastructure. The higher the mobility, the more important the availability of mobile Internet is for coordination.
- Mobile technology: Computing technology is used seamlessly to enable anyplace/anytime work. Different sorts of mobile devices can be used, and the choice is driven by the users' requirements rather than by an organization-wide decision. Mobile devices are always on, are always connected to the Internet, have rapid response rates and reliable connectivity; they are light, small and unobtrusive, and most of all are not prohibitively expensive.
- Management of mobility and mobile working culture: The organization recognizes that mobile teams have different requirements and train their managers to motivate and manage mobile teams. Additionally, organizations are aware of privacy and accessibility issues and develop protocols to help workers maintain a work-life balance.

Within this framework, we propose a solution to common issues of dynamic mobile workflow that exploits the above-mentioned essential features of mobile organizations. We deployed our solution for a simple but insightful scenario, which represent a first step towards the design of more complex mobile processes management systems.

We need to stress here that our contribution is related to the implementation of a mobile workflow and, in particular, that of its user interface, rather than to mobile workflow modeling. Moreover, our scenario

provides us with challenging issues whose solutions can be transferred to real cases.

The paper is structured as follows. In section 2 we present a mobile collaboration scenario and we discuss why kinetic awareness can play a fundamental role in achieving an adequate level of collaboration through an unobtrusive mobile interface. In section 3, we present the Kinetic User Interfaces conceptual framework and its corresponding middleware architecture. In section 4, we present the technical implementation details of the UbiShop prototype. We conclude the paper with section 4 where we establish several future work directions.

2. Mobile collaboration scenario

We consider here a very restricted type of mobile organization, namely a household such as a family or a group of students sharing an apartment. The collaborative task to be accomplished is to ensure that "essential" food (e.g., milk, coffee, beer) is always available, and that whenever somebody planned to cook something for dinner, he/she will find the necessary ingredients when back home.

In order to achieve maximal efficiency, a food manager is required to keep track of what items are missing and have to be purchased. In a cooperative situation, the "food manager" is then responsible to tell other household members to buy the missing items when they go out. This standard solution has several shortcomings:

1. It requires a person being in charge of managing the list of food reserves.
2. Several shopping lists have to be generated at fixed moments of the day (e.g., in the morning when everybody leaves home).
3. It is possible (and sometimes likely) that for dinner not all the necessary items have been bought, because someone was too busy with higher-priority activities, or he/she has simply lost or forgot the list.
4. It may happen that during the day (after the shopping list has been distributed), some item will be consumed and not replaced, or that a member changed his initial plans.

It is easy to see that this situation is not specific to this particular scenario, but applies to a large class of scenarios in which different tasks can be dynamically assigned to workers, who can accomplish them during their mobility.

Standard mobile communication (e.g., via cell phones or SMS) can partially solve this problem. More precisely, it cannot fully solve the fourth issue listed above: the food manager can always contact one of the household members and update the shopping list. However, the manager must take care of not overcharging a single person with too many items to buy. In order to do so, the manager needs to keep track of all the shopping lists that have been distributed. In no case he/she will know who has already bought which items, unless the manager requires to be notified of each item crossed out from the list.

We propose to exploit mobile Internet technology for providing a satisfactory solution to the problem, that addresses all the four issues mentioned above. Our solution is structured as follows:

1. Items are tracked through a bar-code reader or a RFID antenna when they are entered or taken out from the home food container (e.g., the fridge).
2. An overall shopping list is automatically generated from the item database and kept up-to-date by tracking items' in-out activity.
3. The shopping list is always accessible through Internet to household members, that can also add additional items if they plan to cook a specific recipe.
4. Household members are asked to purchase items in the overall shopping list when they have the possibility to do it. In particular, they are requested to buy an item when they are in the proximity of a grocery shop that sells that item.
5. When purchased, the item will be removed from the shopping list, thus avoiding replication. This can be obtained either by the user intervention with a manual deletion from the shopping list, or automatically through a further integration of the system with grocery shop IT services (when the items are electronically scanned at the counter).

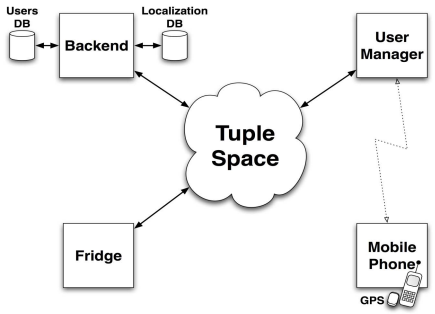

Figure 1. The UbiShop Architecture

It is apparent that this solution solves our problem: a first prototype, called UbiShop [4], has been built on the top of JINI/Javaspace and J2ME technology, whose

architecture is showed in Figure 1. We thus demonstrated that this solution is technically possible with existing technology. In the remainder of the paper we will focus on how such a system can be made more usable with a smarter interaction design.

In the next sections, we describe the KUI middleware architecture and how it has been used for developing the User Manager component and the linked mobile phone of the UbiShop system.

3. Kinetic User Interfaces

We address now the problem of making the mobile interaction with the UbiShop system as unobtrusive as possible. We define *unobtrusiveness* as the measure of how system's interface interferes with the foreground activity of the user. Unobtrusive interfaces are those interfaces that enable the so-called "calm computing" paradigm, envisioned by Mark Weiser in the framework of Ubiquitous Computing [3].

In the UbiShop scenario, we would like to avoid that the user has to continuously check whether his/her current location favors the purchase of an item. This is why we chose to alert the user only when a match exists between the shops available around his/her current location and one or more items to be purchased. In this case, the user is prompted with a partial list of needed items. Hence, it is the location-change that triggers the implicit user query for retrieving the task and checking if the actual contextual conditions are favorable.

3.1 Kinetic awareness

Another important feature of our interface is *kinetic awareness*. A kinetic-aware system is one where motion is used as an input modality. Kinetic awareness can be seen as part of a more general interaction paradigm, defined in [5] as *context-aware computing*. This emerging paradigm poses several challenges. One of the biggest recognized difficulties for this type of systems is interpreting human activity. If, for instance, the user is in a hurry, the mobile interface (e.g. running on the cell-phone) will detect this state by looking at the motion properties of the device. If the speed is detected to be over a certain threshold, it will infer that the user will not be happy to stop at the nearby shop. Hence, it decides to not bother the user with the purchase request.

Similarly, if the user is walking at a reasonable speed, he/she will receive the request. However, if no explicit confirmation is received back within a certain amount of time or if the user keeps walking, the system will interpret this behavior as an implicit refusal to make the purchase.

These two types of input based on kinetic awareness have been implemented and they make the interaction with the UbisShop system less obtrusive. The interaction with the system happens only when strictly required and, in case of unwillingness/impossibility of performing the assigned task, no (explicit) interaction with the user will be done.

3.2 The KUI concepts

The term "Kinetic" derives from the Greek *kinetikos*: "moving of, relating to or resulting from motion (the action or process of moving)". In physics, kinetic theory explains the "physical properties of matter in terms of the movement of its constituent parts"; kinetic energy refers to "energy, which a body possesses by virtue of being in motion". Kinetic abilities of humans are of no question. People move and change their current spatial location all the time and in a mostly unconscious way. Humans are also capable of "modulating" motion in several ways, by keeping or varying their speed, by following different trajectories or patterns (e.g., dancing), or by executing various types of motion in parallel (e.g., gesturing while walking). At different scales and contexts, motion (or absence of motion) can be recognized as a purposeful action.

We introduced the concept of *Kinetic User Interface* (KUI) in [6] as a way of endorsing the Weiser's Ubiquitous Computing vision [7] and the Dourish's Embodied Interaction vision [8]. Accordingly, KUIs are intended to enable a new interaction model for mobile computing systems in which motion of objects and users in the physical space are recognized as events and processes to which the system reacts. To put it in terms of Instrumented Interaction [9], the space becomes an *instrument* of the user interface and motion is one of its afforded *actions*.

Similarly to "hovering" the pointer over a desktop in GUIs, in KUIs users can trigger input events for the computing environment by moving themselves and by displacing tracked objects. Users can exploit the physical space by executing actions/operations on physical objects, such as moving, grabbing, touching, juxtaposing, whose effects are reflected in the application domain objects.

KUIs are not limited to single-user interfaces and do not impose a unique locus of interaction. Hence, it enables richer interactions than GUIs and thus better suited for ubiquitous mobile computing environments. The kinetic input modality can be used alone or in combination with other input modalities available to the user for interaction with the system, which are directly afforded by other mobile devices carried by the users and by fixed input devices located in the

interaction space (e.g. ordinary point and click, or speech recognition).

3.3 KUI interaction patterns

KUI enables several interaction patterns. We have already discussed one of them presenting the UbiShop scenario. With KUIs, it is possible to transfer common GUI interactions in the physical space such as Drag&Drop. For instance, in a KUI-enabled SmartHome, the user can "drag" the media being currently played in the living room and "drop" it to the bedroom just by moving a representative localizable object such as the remote controller.

Another useful KUI pattern is *continuous tracking*. Continuous physical motion is comparable to "mouse-gestures". KUI-enabled applications are supposed to recognize certain *kinetic patterns* that might be naturally performed by users during other activities or specific situations. As an example of the this pattern, consider a scenario where the user is driving a car and some of the car's motion parameters are obtained by embedded sensors such as a GPS tracking system and an accelerometer. The sensors reveal that the car is decelerating in the proximity of a gas station (i.e. a geo-located point of interest already known by the application). The KUI-based application detects and interprets deceleration as the user's intention of refueling at the gas station. This hypothesis might be corroborated by other contextual information from the current car's sensors (e.g., the fuel level being almost zero). As a result of this behavior, the system will pro-actively prompt the driver with the current gas prices at the approaching gas station. The application might also perform further contextual inferences and inform the user that keeping the current speed and considering the current fuel level he/she can reach the next gas station that has better gas prices. However, if the system detects that the fuel level is high or the fuel tank is even full, it will not react because it can infer that the driver stops for other (unknown) reasons (e.g., to take a break).

3.4 The KUI middleware architecture

In this section, we present the main concepts of KUIs that are implemented as software components in the KUI middleware architecture.

In KUI, motion is a main (or primary) interaction modality afforded by the physical space to users through the motion of *tracked entities*. Tracked entities are any objects or autonomous (possibly living) things for which we can provide location and motion information. Tracked entities are represented by KUI components called *Kuidgets*. Interaction with Kuidgets happens when users affect their motion properties or change spatio-temporal relationships between some of them (e.g., an object is entering into an area). For instance, when the user is driving a car the motion properties of its corresponding Kuidget will be continuously updated with its current position, speed, acceleration, and direction.

Two Kuidgets can be logically linked and one of them can provide location and motion information to the other. For instance, a car equipped with a GPS sensor (an artefact Kuidget) can provide location and motion information to its driver (an agent Kuidget). Vice versa, a user carrying a GPS sensor can provide location and motion information to the car Kuidget by setting a logical link between the car and the user who is driving the car. It is important to notice that when the user leaves the car, even if the link is destroyed, the car Kuidget keeps its last location.

The detection of particular spatio-temporal relations between Kuidgets can trigger application-specific KUI events. There are several spatio-temporal relations that can be modeled. We propose here that a basic KUI should provide at least two types of spatio-temporal relations: *proximity* and *containment*. For these relations it is important to consider their temporal dimension, namely the start and end time, and the duration of the relation. For instance, if two mobile Kuidgets (e.g., two agents) are observed while moving together along the same path or into the same location, the application will be notified with a "joint motion" event by the KUI manager. Then, the application might make inferences, and as a result, establish an application-dependent relation between the two Kuidgets. For instance, when the two agent Kuidgets are moving together, the application can infer (with the help of other contextual information) that they might be friends. Analogously, when two Kuidgets expected to move together, start moving in different directions, an event could be triggered that in certain circumstances could denote an unusual situation. For instance, a car moving while its owner moves somewhere else might denote that the car has been stolen.

3.5 The KUI middleware

The KUI middleware is made of three layers of abstraction (as shown in *Figure 2*):

- The *Observation Layer* is the lower level. Its role is to collect kinetic contexts from location and motion-aware devices and from other hardware sensors. Hardware motion-aware sensors are encapsulated by re-usable components called *Widgets*.

- The *KUI-Space Layer* is an object-oriented environment that contains and manages Kuidgets state and their semantic relationships. Information flow coming from the observation layer is used to update the state of Kuidgets. Location information pertaining to Kuidgets is stored in a data structure (the GeoDB) that is also used to store and manage physical space references of fixed and mobile Kuidgets for both topological and symbolic spaces.
- The *Activity layer* manages the context history, and aggregates KUI events into higher-level semantic events (for instance, for the detection of specific interaction patterns) that are sent to the applications. In this layer representations of *activities* will be constructed by aggregating kinetic information from Kuidgets and other contextual information. Moreover, models of activities will be matched with the spotted situations in order to determine the occurrence of anomalous behaviors.

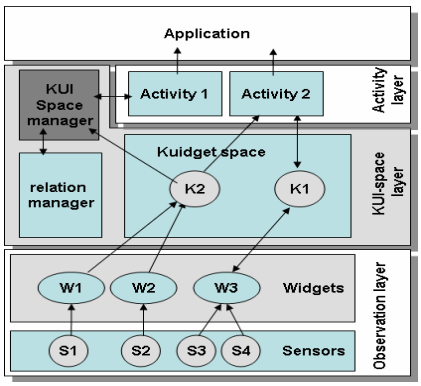

Figure 2. The KUI middleware architecture

More specifically, in the KUI software components location and motion information are linked to either fixed Kuidgets or mobile Kuidgets that are localizable by means of tracking hardware. In the UbiShop system, all layers are deployed on the backend server (i.e., the User Manager module), while sensor widgets run on the mobile device. It is also our goal to make KUI as general as possible in order to uniformly cope with different geographical scales (e.g., tabletop, room, building, cities) and with different types of motion-aware devices (e.g., GPS, RFID, Wireless cell triangulation, ultrasonic, ultra-wideband, infrared). For this purpose, we also distinguish between different *types* of physical space.

3.6 Related work

Kinetic-awareness is not a completely new concept since it has been already implicitly adopted in several projects (see [5], [10] and [11] for a survey). Nevertheless, the novelty here is that we propose a conceptual and software framework for the rational design of this type of interfaces.

The applicability of the KUI framework has been preliminarily assessed in [6] and some scenarios have been classified in [12].

4. The UbiShop prototype

UbiShop [4] is a kinetic-aware mobile and distributed application based on the KUI middleware. The KUI-middleware is deployed both on a server application and on mobile, networked handheld devices (J2ME enabled cell phones).

4.1 Architecture

The KUI-Space Manager and the Activity layers reside on a server (together with other UbiShop modules as shown in Figure 1) and constitute an essential part of the User Manager module. The mobile components of UbiShop are also based on the KUI middleware, namely on the Observation layer. The cell phone client encapsulates a Location Widget linked to the User Kuidget through a mobile Internet connection (e.g. GPRS or UMTS) as shown in Figure 3. The Location Widget wraps a GPS antenna (integrated or connected to the cell phone through Bluetooth). The Widget sends GPS coordinates to the server, which in turn updates kinetic information for the corresponding Kuidget.

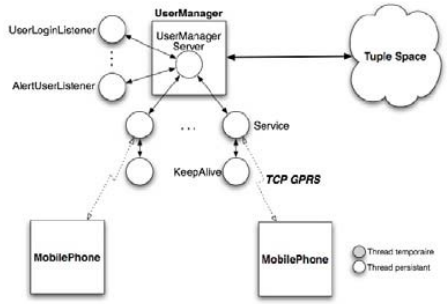

Figure 3. Connections between the UserManager module and Mobile Clients

The Activity layer detects when a particular spatial relation is created, for instance when the user is passing by a relevant location (e.g. a grocery store) with an acceptable speed (e.g. below 5 Kmph). In that case, the User Manager is notified of this event by the Activity layer, and it consults the JavaSpace to check if there is any item from the shopping list that can be purchased at the current user's location. If some items are available, then the User Manager alerts the user by sending a purchase request and waits for an answer. The user can either interact with the client's GUI or

ignore the request. In the latter case, the User Manager looks at the KUI-Space and checks whether the User Kuidget is moving or not. If the user is not moving, then it waits for an explicit answer. Otherwise, after a timeout, an implicit refusal to make the purchase is inferred.

If an item has been purchased and requires to be put rapidly in the fridge, UbiShop ensures that this happens by notifying the Fridge component that the user is back home. More technically, when a perishable food item is purchased, the Fridge component is informed and it subscribes for the event "user is back home". When the user is back home, the User Manager informs the Fridge of this event and the Fridge starts a timer. When time is over and the perishable item is not yet in the Fridge, the Fridge asks the User Manager to remind the user to put the perishable item in the Fridge. This protocol is implemented through asynchronous communication (using JavaSpace).

4.2 Performance and Evaluation

The UbiShop prototype has been tested on a small scale (campus-wide) and it shows a fairly robust behavior. Since it uses asynchronous communication, if a module crashes or Internet is not available, the processes can be interrupted and resumed when the system consistency is reestablished. Moreover, the components are loosely coupled and the system can run also when not all the modules are available.

We plan to perform a larger scale evaluation, possibly with the support of a local mobile telephony company in order to test the users adoption of this technology as well as its usability.

5. Conclusion

In this paper we have addressed the problem of mobile collaboration and coordination from the perspective of the user interface. We have presented a collaboration scenario for a mobile organization that requires a degree of flexibility, which can only be obtained through a smart use of mobile Internet technology and location/motion awareness. We have developed a functional prototype that implements the Ubishop scenario based on the Kinetic User Interface (KUI) framework. The KUI framework is under development at our institution and offers the support for several kinetic interaction patterns. We believe that kinetic user interfaces will play a central role in the next generation of mobile interfaces, as they provide users with the right level of unobtrusiveness. Moreover, the growing market of positioning devices [13] and motion sensors (e.g., the WiiMote) makes this type of systems possible and reliable.

5.1 Future Work

Within our framework, we are exploring two main directions of future work. First, we are improving the middleware in order to provide a better integration with ad-hoc networks created on-the-fly between local mobile devices. This would avoid having a unique point of centralization (and thus of failure) and would make it truly distributed. Second, we are working on end-user programming in order to allow non-expert users to visually design their KUI-based applications. For instance, they will be allowed to mark zones on a map and tell which services are triggered against the creation/destruction of spatial relationships with Kuidgets.

6. References

[1] D. Neal. *Collaboration – The key to getting value from new mobile technologies*, CSC Research Services Web Conference, www.csc-researchservices.com, 2003.

[2] H. Schaffers, T. Brodt, M. Pallot & W. Prinz. The Future Workspace. *Mobile and Collaborative Working Perspectives*, Mosaic consortium, Telematica Instituut, The Netherlands, 2006.

[3] M. Weiser and J. Seely Brown. *The Coming Age of Calm Technology*. Technical Report Xerox PARC October 5, 1996.

[4] T. Maret, N. Martenet. *iFridge*. Project Report, Dept. of Computer Science, University of Fribourg. 2007.

[5] A. Dix, J. Finlay, G. Abowd, & R. Beale. *Human-Computer Interaction* (Third Edition). Prentice Hall. 2004.

[6] V. Pallotta, P. Bruegger & B. Hirsbrunner. Kinetic User Interfaces: Physical Embodied Interaction with Mobile Pervasive Computing Systems. In Kouadri-Mostéfaoui, Maamar, Giaglis (eds.), *Advances in Ubiquitous Computing: Future Paradigms and Directions*. IDEA Group, 2007.

[7] M. Weiser. *Hot topic: Ubiquitous computing*. IEEE Computer, pp. 71-72, October 1993.

[8] P. Dourish. Where the Action Is: The Foundations of Embodied Interaction. Cambridge. MIT Press. 2001.

[9] M. Beaudouin-Lafon. Instrumental Interaction: An Interaction Model for Designing Post-WIMP User Interfaces. In Proceedings of CHI 2000 conference. 2000.

[10] G. Chen & D. Kotz. *A Survey of Contex-Aware Mobile Computing Research*. Tech. Rep. n. TR2000-38, Darthmouth Science Department, 2000.

[11] M. Baldauf, S. Dustdar & F. Rosenberg. A Survey on Context Aware Systems. *International Journal of Ad Hoc and Ubiquitous Computing*, Inderscience Publishers. Forthcoming, 2007.

[12] T. Maret. *A* Classification Scheme for Kinetic User Interfaces. *PAI Working Paper*, Department of Computer Science, University of Fribourg, 2007.

[13] J. Hightower & G. Borriello. Location systems for ubiquitous computing. *IEEE Computer* 34(8), 57-66, 2001.