A Graphical User Interface for the Real World

Kasim Rehman
Laboratory for Communication Engineering
Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
United Kingdom
kr241@cam.ac.uk

ABSTRACT
A number of typical Ubiquitous Computing (Ubicomp) interface problems are presented. From these, the requirements needed for a Ubicomp meta-user interface are extracted and a user interface architecture that is based on traditional GUI architectures is introduced.

MOTIVATION
A recent survey of more than 100 Ubiquitous Computing Applications (Rehman 2001) revealed that a lot of these applications violate basic HCI guidelines such as Norman's design principles (Norman 1990). More specifically, developers often leave the user with too little control, don't provide appropriate feedback about what the system is doing and fail to show appropriate affordances and constraints to the user. A more detailed analysis is given in (Rehman et al. 2002). For our purposes I shall try to highlight the fundamental problem here.

My hypothesis is that these problems are not singular design flaws, but have their root in the very idea of Ubiquitous/Disappearing/Invisible Computing. First, there is a need to define "invisibility". Some system developers have taken it to mean that the user does not see any system-related part physically. As pleasing it may seem to some developers, this is not what Weiser or Norman meant when they conceived the idea, as is evident from the examples they usually cite. Rather, they meant that the computer stays out of the user's mind, not necessarily their sight. That said, let us look at whether even this is achievable.

For this we shall look at the archetype of a good tool: the pencil. You have control over it, the user knows how to use it, and it provides constant feedback. More importantly, it is invisible in Weiser's sense (Weiser 1994). I believe that the reason why we have not achieved this quality in our applications is, that there is an asymmetry in our capability to capture input in comparison to providing output. With the advent of sophisticated sensors any real world object, even the human body (as seen in location-aware applications) can be a highly accurate input device. Whereas the above-mentioned pencil can also deliver feedback in the same way as its input (mechanically), we cannot, say, mechanically stop a museum visitor with a context-aware electronic guide from going out of the coverage of the location system.

One solution is not to worry about the output at all. Clearly, this leads to ill-designed applications. The other solution is to display this information on a display. This adds another level of indirection a conventional tool does not suffer from and, depending on where the display is placed, may imply an increase in cognitive load.

Before we talk about our approach to this problem, we shall look at another problem we became aware of when analysing the applications. Norman (1993) talks about the idea that an interface in its function as a "surface representation" should convey an image of the underlying system. The problem we have in Ubicomp the system is a collection of invisibly connected heterogenous nodes that do not seem to have a "surface" a user can relate to, at all.

Our aim is to not only give the user such a smooth "surface", but also to massively increase the bandwith of the reverse channel, between system and user. In order to do this with as little cognitive load as possible, we advocate the use of Augmented Reality (AR).

SYSTEM IDEA
In its widest sense any system that connects the real and virtual world can be labelled "Augmented Reality" (AR). As such, even tangible interfaces are examples of AR. The narrower definition involves a system that uses a head-mounted display (HMD) and a tracker (Feiner et al. 1993). The tracker continuously measures the position and orientation of the head to some real object and displays a 3D graphics on a see-through HMD that makes the virtual object appear to be placed at a fixed location in the physical world. We are using a head-mounted camera and have deployed markers (Kato & Billinghurst 1999) in order to track objects of interest. The marker is a reference point that can be used to overlay graphics on a real world

object. A file cabinet can, e.g., have its contents overlaid on it, as long as the marker has a fixed relationship to the file cabinet. The system will infer the position of the file cabinet relative to the user's eyes and display a corresponding virtual object on his HMD.

In order to design our smooth surface, we want to leverage some of the design principles used in the GUI domain. Before GUIs arrived, computer users were typically confronted with a multitude of applications each with their own user interface. Transferring data between them was difficult and users had to interrupt their tasks in order to adhere to application boundaries. Working across applications was impossible. In a way the situation was similar to the one we now have in Ubicomp. The arrival of a meta-interface was decisive in giving the user the "unified experience" we wish to provide in our present domain.

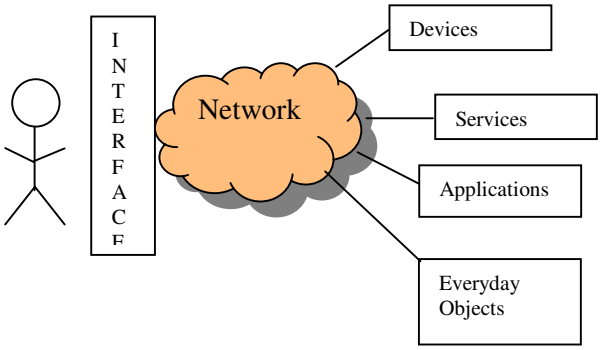

Figure 1: A meta-interface for ubiquitous computing

A VDU (VISUAL DISPLAY UNIT)

The first step in building our Graphical User Interface that can send information and receive information from any everyday object, service, appliance or application, was to implement a display that covers the entire space. Of course, this is meant in a virtual sense, using Augmented Reality. This involved constructing a spatial model that abstracts from tracking sensors and sources.

We are working with a spatial model that consists of a network of points. The arcs are Cartesian coordinate transformations, some dynamically changing, some fixed. We are using this type of model in order to cope with 6-DOF(degrees-of-freedom) tracking information. In order to give an object output capability, say a loudspeaker, we can attach an electromagnetic sensor or a marker to it and add it as a point of interest to our model, that will keep track of its position and orientation at all times.

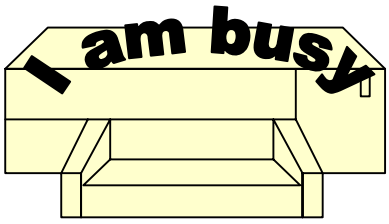

Figure 2: Every object has an output

USER INTERFACE ARCHITECTURE

We shall now describe the entire user interface architecture we are in the process of implementing. Figure 3 shows the current design plan. We are assuming that the user has a mobile unit with a head-mounted display and that his head position and orientation are trackable. The environment contains a number of "active" appliances: devices such as a printer, services such as a web search, Applications such as Microsoft Word and Everyday Objects such as a mug that knows its temperature.

All of these devices have interfaces to a tuple space in order to receive commands and send messages to the user. We thought that a tuple space is best suited for a symmetric 2-way communication. The two paths of information flow are shown by the arrows: a forward path from left to right and a feedback path from right to left.

The world model and data model can be seen as the Visual Display Unit described above. The world model contains the position(and orientation) of each active object. The data model is a description of interactive information, each piece of which

is attached to some point defined in the world model. Such a piece of information can be a piece of text, an error message, a choice list with associated actions, a description of a graphical object etc. The important thing is that *all* messages for the user shall be relayed through the feedback component to the data model.

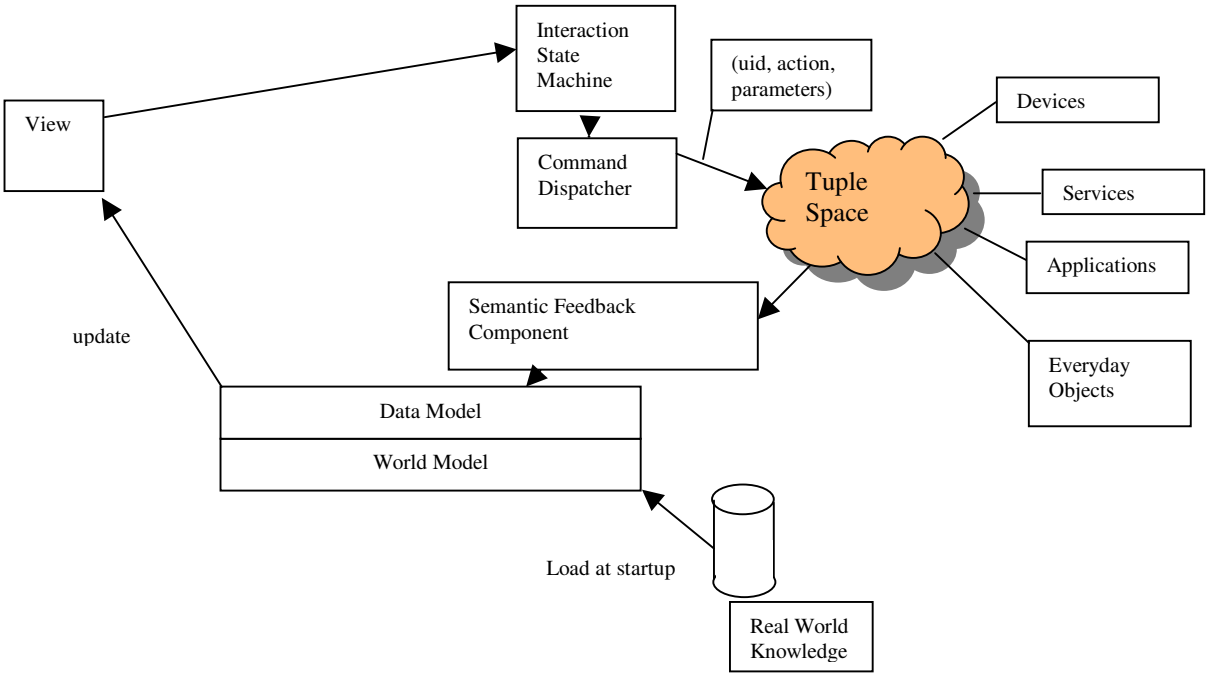

Figure 3: User Interface Architecture

The view component at the user's mobile unit reads in the model and creates a graphical representation for the user. It also takes care of providing intermediate feedback during interactions. Interactions take place by using a 6DOF input device with which virtual representations of the objects can be selected. All virtual objects and hence the real objects are clickable. While this is taking place an interaction state machine changes state according to what action is being performed, e.g. dragging, clicking. A command dispatcher sends a command to the tuple space, as soon as it knows what command is to be performed. A typical action could be following: The user clicks on the virtual object overlaid on a CD, the system finds out its unique identifier by using the world model, since all unique identifiers have been loaded into the world model at startup. As she drags the object towards the virtual representation of the CD player, the view provides continuous feedback. As soon as it is dropped into the CD player the command dispatcher sends a command containing the CD player's unique identifier, the CD's identifier and the action name.

The key to the success of Ubicomp will be synergy: The combination of capabilities of devices that have never met in order to facilitate user tasks. Therefore, what is needed is a user interface that encourages synergy. In order to achieve this more work has to be done interfaces between the tuple space and the active objects. The advantage of using a tuple space is that have never met can synchronise. They can become aware of what other devices are doing and can send messages according to that. Whenever the user picks up a data item, services and devices that are ready to accept it can signal so by sending a message to the user, which may ultimately transformed into a visual cue.

CONCLUSION

In this position paper I have presented my motivation for constructing a user interface architecture that provides for information flow in both directions. Given our current design plan, I have demonstrated that such an implementation is feasible. I have also shown that a meta-interface can provide facilities, a number of heterogeneous user interfaces do not provide. So far, we have actually implemented the world and data model.

REFERENCES

Feiner S., Macintyre B., Seligmann D. 1993, Knowledge-based augmented reality, vol. 36, pp. 53-62, *Communications of the ACM*.

Kato H., Billinghurst M. 1999, Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System, pp. 85-94, *Proceedings IWAR'99*.

Rehman K. 2001, *101 Ubiquitous Computing Applications*, Available: http://www-lce.eng.cam.ac.uk/~kr241/html/101_ubicomp.html.

Rehman K. 2002, Interfacing with the Invisible Computer, To appear *Proceedings NordiCHI 2002*, Available: http://www-lce.eng.cam.ac.uk/~kr241/Paper260702.pdf

Weiser M. 1994, *Building Invisible Interfaces*, Presentation Slides, Available: http://nano.xerox.com/hypertext/weiser/UIST94_4up.ps