A Reference Framework for Multi-Surface Interaction

Christophe Lachenal *Joëlle Coutaz*

CLIPS-IMAG Laboratory, University of Grenoble
Domaine Universitaire, BP 53, 38041 Grenoble Cedex 9, FRANCE
{Christophe.Lachenal, Joëlle;Coutaz}@imag.fr

Abstract

Physical surfaces are pervasive and serve many purposes. Digital computation is a powerful source of functional support. However, it has been confined to the augmentation of single objects only. In this article, we are interested in the combination of physicality with computation in the context of multiple objects. We propose the notion of multi-surface interaction as a unifying paradigm for reasoning about both emerging distributed UI's and known interaction techniques such as GUIs, tangible UIs, and manipulable UIs. Multi-surface interaction is expressed within an ontology that shows how our concepts feed into the design of sound foundational software for the development of multi-surface user interfaces.

1 Introduction

Surfaces play a predominant role in our daily activities: in civil architecture, they structure the space into places. Any artefact has a shape that can be comprehended via the surface it offers. In computing, display screens and electronic boards are surfaces used for rendering information. Because digital computation is a powerful means for "crunching" information, it is interesting to bring together physical surfaces with digital information. By doing so, we obtain information surfaces. However, information surfaces are useful for humanity only if humans can manipulate them. Manipulation coupled with observation form the essence of interaction. By which we justify the term "multi-surface interaction".

Recent advances in HCI show many examples of user interfaces based on the simultaneous use of multiple surfaces: I-land (Streitz & al, 1999, p. 553), iRoom (Johanson & al, 2002), or Rekimoto's augmented surfaces (Rekimoto & Masanori, 1999, p. 378) and Data Tiles (Rekimoto & al, 2001, p. 269) are typical examples of multi-surface interactive environments. From centralized display on a single surface, user interfaces can now migrate freely and be distributed across multiple surfaces. For example, proximal PDA's could provide a reconfigurable mosaic on which the user interface can expand and shrink dynamically. Although development tools have made significant progress, they are still limited in a number of ways to support the implementation of multi-surface user interfaces. For example, windows are modelled as rectangular drawables whose borders are constrained to be parallel to that of the display. This model is based on the (wrong) assumption (for ubiquitous computing) that users keep facing a vertical screen and use a single pointing device at a time. This assumption needs to be revised leading to new requirements.

In this article, we propose an ontology that clarifies the notion of multi-surface interaction. In turn, the ontology calls for the development of new software foundational infrastructures such as I-AM. I-AM is an Interaction Abstract Machine whose level of abstraction is similar, in spirit, to that of windowing systems such as X-Window. In addition, it supports the dynamic reconfiguration of the underlying physical infrastructure, as well as the dynamic distribution of user interfaces across multiple interaction resources.

2 Ontology

Figure 1 shows a UML formal description of our ontology for multi-surface interaction. The components of the ontology include: Interaction resources that serve as mediators between an artificial actor (e.g., a ubicomp system) and a natural actor (e.g., a user). An interaction resource may serve as an instrument or as a surface. As an instrument, an interaction resource mediates the actions of an actor. As a surface, the outmost boundary of a physical artefact serves as a recipient for making information observable to an actor. Actors have actuators to modify the state of an interaction resource, and have sensors to observe the state of interaction resources. Actors and Interaction resources are members of a topology (i.e., spatial relationships). Interaction resources and their relationships have attributes (e.g., size, social use) and properties (e.g., maniability) that can be used to characterize a particular multi-surface interaction.

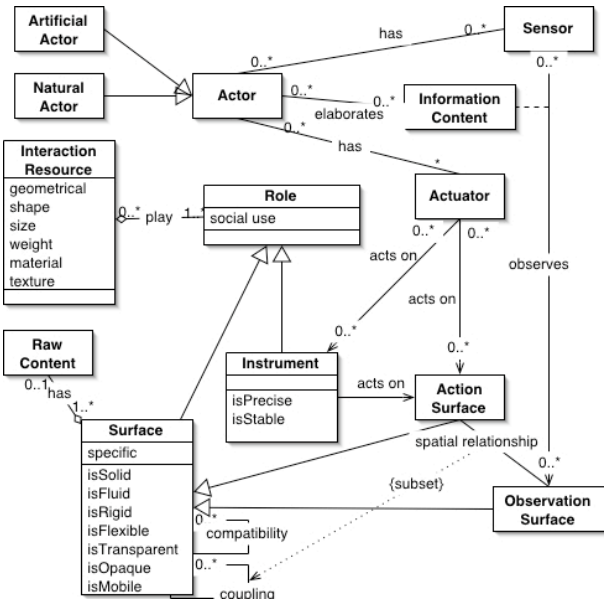

Figure 1 : **UML description of the Multi-Surface interaction ontology.**

For example, in the GLOSS multi-surface system shown in Figure 2, a laser pointer is, from the user's perspective, an instrument to select raw content displayed on two surfaces: a wall and a table. The laser light is detected by the system using a camera sensor. The pucks of the Senseboard, on the other hand, serve both as instruments and surfaces (Jacob et al, 2002, p 246). The Senseboard provides a tangible user interface for manipulating and organizing pieces of information as in spreadsheets. Each puck is a small magnetized plastic tag that denotes a piece of information. It can be placed into grid cells to move information. As such, it is an instrument for the user. In addition, a video-projector displays specific information on the top of the puck: it also serves as a surface that the user can observe.

The relations "acts on" and "observes" call for the definition of two classes of surfaces: action surfaces and observation surfaces. An action surface is a subset of a physical surface on which an actor can act directly with actuators and/or indirectly with instruments. Similarly, an observation surface is a subset of a physical surface that an actor can observe with sensors. For example, the top of a Senseboard puck is an action surface for the system and an observation surface for the user. The adequacy of a surface for action and/or for observation depends on its attributes (such as size, weight, and material) and properties (e.g., fluidity, flexibility, opacity, transparency, etc.).

By composing surfaces in space, we build geometric configurations. In turn, the geometric configuration sets the foundation for analyzing properties of the spatial relationships between surfaces. Typically, an empty intersection between the action and the observation surfaces of a human actor, predicts discontinuity (e.g., the user can't see what he is doing). In addition, the way surfaces are coupled to form a geometric space opens a rich line of reasoning. For example, in modern classrooms, blackboards are comprised of several panes that can be moved up and down as needed. Teachers frequently project prepared slides on the boards. They augment the slides opportunistically by writing on the board with ink-pens. When they scroll the board, the ink inscriptions are scrolled but the projected slide is not. In this setting, the system, which has no sensing capacity, is unable to adjust its action surface to that of the user.
Similarly, spatial relationships of the surface configuration with regard to the user's position matters. For example, in Rekimoto's augmented surfaces, objects are represented as 3D graphics interactors on laptops, whereas 2D rendering is used for objects placed on a horizontal surface. In these examples, the orientation of the rendering surfaces relative to the user (e.g., "horizontal" and "vertical") determines the nature of the output modalities. So far, we have illustrated the ontology with examples drawn from the user's perspective. In the following section, we describe the consequences from the system perspective.

3 A Foundational Software Platform: I-AM

The goal of I-AM is to provide developers with a uniform logical space composed of any number of processors, and any number of interaction resources and classes of interaction resources. Because interaction resources can be "plugged" and "unplugged" at will, I-AM must include mechanisms for resource discovery. Interaction resources tend to go wireless and can easily be moved around. Therefore, proximity detection and orientation are required to maintain an operational topology (e.g., the orientation of interactive surfaces matters!). A first version of I-AM is under development illustrated with the implementation of a multi-surface user interface distributed across a wall, an augmented table, and a PDA.

3.1 An illustrating example: the GLOSS multi-surface setting

In order to understand the services that I-AM should provide, we have developed a multi-surface prototype system. As shown in Figure 2, the user interface is distributed across a wall, an augmented table, and a PDA.

Figure 2: **The GLOSS multi-surface setting: a large public surface (a wall), a semi-private surface (a table), a private surface (a PDA).**

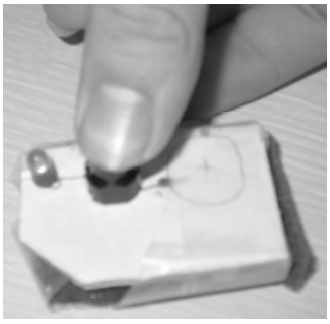

Figure 3: **The GLOSS clicker.**

The prototype was developed with the idea that it could be placed in a bar with a high level of integration in the environment. In the bar, many surfaces can be used: tables to put glasses, walls to display menus, etc. Our idea is to exploit the physical surfaces of a bar and augment them with computational facilities such as sending digital postcards or reading SMS. As

shown in Figure 2, the wall, which is large public surface, is used to display six stacks of postcard. Each stack represents a theme (e.g., Paris by night, Monuments in Paris). Below the stacks of postcards, the system shows a thumbnail of available physical tables. Users, sitting at a table, can select a stack from the distance using a laser pointer. Dragging a stack with the laser beam to the thumbnail table, brings the postcards of the stack on the table. Each circular table is used as a semi-private surface on which the user can select a postcard, write on it and send it. If the user has a PDA, he can also save postcards on it. To do that, the PDA and the table have to discover each other.
Because the table is circular, the user interface can be conveniently rotated towards the user using the wireless GLOSS clicker. A shown in Figure 3, the GLOSS clicker has been built from a press button and a LED mounted on top of a 2x4cm foam box wrapped with paper. The LED is tracked with computer vision. In addition, the user can receive SMS messages on the wireless PDA. The display screen of a PDA is rather small for editing messages. Therefore, the surface of the table can be used to read and write messages. As a result, the table serves two types of activities: browsing and writing postcard, viewing and writing text messages.
This prototype shows that building multi-surface user interface is not straightforward: the programmer needs to know which surfaces are available in the space, what are their characteristics, what are the spatial relationships between the interaction resources, and so on. We have designed I-AM to alleviate programmers development effort.

3.2 I-AM

I-AM is a software infrastructure that extends the functional coverage of current windowing systems to multi-surface interaction. As shown in Figure 4, it is comprised of four levels of abstraction. At the lowest level of abstraction, the network layer is in charge of interaction resource discovery as well as of their communication. As specified in our ontology, interaction resources appear and disappear from the interaction space (ex: a guest sits at Table 3, a PDA is close to Table 5, a new table is available). Because multiple interaction resources may be connected to a single computer, network protocols like UPnP or Zeroconf could be used to discover them. Such protocols have been developed to allow devices with network capabilities to discover each other without any network administration. Although these protocols are adequate for medium scale networks, they do not support any information about the network topology nor about the resources topology.

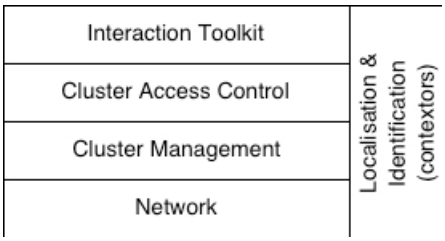

Figure 4: **I-AM software architecture layers decomposition**

As shown in previous Section, topology is central to multi-surface interaction. Topology modelling relies on a service capable of localising and identifying resources. The cluster Management layer provides the programmer with a logical view of interaction resources. Whereas a windowing system manages windows (i.e., logical screens), a single pointing device and typing device, the cluster level manages a dynamic set of surfaces and instruments, along with their geometric configuration in space, and their coupling. For example, in the GLOSS prototype, the wall, and the tables form a cluster of interaction resources. This cluster evolves over time: the topology of the tables may change (two tables put side by side form a larger surface to interact with. When a client sits at the table, the cluster may be enriched with

the PDA interaction resources. From the programmer's perspective, this situation translates to subscribing to events such as "new surface has arrived" and binding these events to specific actions. On top of the Cluster Management, the Access Control layer supports the capabilities for joining/leaving the cluster. For example, the "bar cluster" may not allow any PDA to join the interaction space. At the top of the hierarchy, the toolkit layer corresponds to the levels of abstraction of the current "graphical" machine and the widget toolkits. However, this layer needs to address the dynamic adaptation of the rendering to physical surfaces and instruments. For example, in the GLOSS setting, the graphical output must be mapped onto a circular surface. As a result, the graphical user interface should fit perfectly the shape of the disk, and should be rotative. In addition, if the table cloth were green instead of white, color rendering should be changed accordingly to maintain readability. Sensing the color of the table (i.e., a surface characteristics) is performed with the sensing contextors (Crowley & al, 2002, p. 117).

4 Conclusion

The emergence of ubiquitous computing calls for the definition of new conceptual frameworks for reasoning from both the user and the system perspectives. Our notion of multi-surface interaction is an attempt in this direction. Our ontology for multi-surface interaction makes explicit the following important concepts: the dual view, surface-instrument, that any interaction resource may support; the distinction between action surfaces and observation surfaces; the symmetrical role between natural and artificial actors, both of them being characterised by actuators, sensors and information content; the spatio-temporal coupling with content; and the dynamic configuration of interaction resources and actors within a topology. Topology along with a unifying software infrastructure that manages dynamic heterogeneous clusters of interaction resources, actuators and sensors, are the next software challenge to be addressed.

5 Acknowlegments

This work has been partly supported by the IST-FET GLOSS project (IST-2000-26070) and IST FAME project (IST-2000-28323).

6 References

Crowley J. L., Coutaz J. , Rey G., Reignier P (2002). Perceptual Components for Context Aware Computing. In *Proc. of the ACM conf. Ubiquitous Computing (UBICOMP),* 117-134, Springer.

Jacob R., Ishii H., Pangaro G. Patten J (2002). A Tangible Interface for Organizing Information using a Grid.. *In Proc. of the ACM conf. On Human Factors in Computer Human Interaction (CHI)*, 2002, 339-346. ACM Press.

Johanson B., Fox A., Winograd T (2002). The Interactive Workspace Project: Experiences with Ubiquitous Computing Rooms. *In IEEE Pervasive Computing Magazine*, 1(2), April-June 2002.

Rekimoto J., Masanori S (1999). Augmented Surfaces : A Spatially Continous Workspace for Hybrid Computing Environments. *In Proc. of the ACM conf. On Human Factors in Computer Human Interaction (CHI)*, 378-385. ACM Press.

Rekimoto Y., Ullmer B., Oba H (2001). DataTiles: A Modular Platform for Mixed Physical and Graphical Interactions. *In Proc. of the ACM conf. On Human Factors in Computer Human Interaction (CHI)*, Seattle, 269-276. ACM Press.

Streitz N., Tandler P., Müller-Tomfelde C. , Konomi S (2001). Roomware: Towards the Next Generation of Human-Computer Interaction based on an Integrated Design of Real and Virtual Worlds. *In Human-Computer Interaction in the New Millenium*, 553-578. Carroll J. (Ed.), Addison-Wesley.