A Visual Language for Programming Reality-Based Interaction

Orit Shaer and Robert J.K. Jacob
Computer Science Department, Tufts University
{oshaer, jacob}@cs.tufts.edu

Abstract

Reality-based interfaces (RBIs) offer the promise of natural interfaces that are intuitive to use. By allowing users to leverage their innate skills and knowledge about the real-physical world, RBIs could benefit disadvantage populations that rely on physical I/O modalities and tangible representations. We present our research which is aimed at simplifying the development of RBIs by providing a technology-independent visual language for modeling and programming these interfaces. Such a language will enable developers to systematically analyze and compare interaction designs as part of the development process while addressing issues such as users' skills and needs. Ultimately, we expect this research to lower the barriers for developing RBIs.

1. Introduction

The last decade has seen a wave of HCI research that led to the development of a range of interaction styles such as tangible user interfaces (TUIs) [1], ubiquitous computing, and physical interfaces. All of these extend beyond the limitations of a two dimensional display, a mouse and a keyboard, to change interaction with computers from an isolated activity to one that is taking place within the real-physical world and is hence similar to daily real world activities. By making interaction with computers more like interaction with the real-physical world, these interfaces allow users to leverage their existing skills and expectations from the real physical world while accessing and manipulating information. Thus, we refer to these interfaces as Reality-Based Interfaces [7] (RBIs).

RBIs embed computation in the real-physical world and often employ metaphors that give physical form to digital information [1]. A reality-based interaction takes place within the real-physical world and typically employs manipulation of artifacts or performance of gestures. As a reality-based interaction takes place within the physical world, users are allowed to engage their full bodies in the interaction. Furthermore, such interaction often involves multiple users interacting in parallel with multiple devices. Because RBI's leverage users' existing skills, and real world knowledge as well as giving tangible representation to digital information, they offer the possibility of natural interfaces that are intuitive to use. Thus, RBIs could especially benefit populations with special needs such as children [3, 6] and elderly [4].

Although RBIs offer the possibility of interfaces that are easier to learn and use, they are currently more difficult to build than traditional interfaces. Current event-driven software models fail to explicitly capture aspects of realty-based interaction; examples include physical representations, parallel digital and physical output channels, and concurrent interaction of multiple users. Thus, RBI developers face challenges such as analyzing, comparing, and communicating alternative interaction design. Furthermore, the lack of software toolkits aimed at RBIs requires developers to deal with physical I/O using low-level programming. Among the technologies commonly used for physical I/O are computer vision, RFID, and microcontrollers. Also, mobile devices such as PDAs and wearables are often used to build RBIs, As each of these technologies currently requires a different set of physical devices and instructions integrating and customizing them to an RBI application is difficult and costly.

Our research aims to simplify the task of building RBIs by providing developers with a technology-independent, high level, description language that would enable developers to easily analyze, coherently discuss, and rapidly implement RBIs. To meet this goal, this research is designed as an iterative cycle which consists of three stages: 1) Identifying a set of core constructs for describing the structure and behavior of RBIs 2) Developing a modeling language 3) Evaluating, reflecting, and redesigning.

2. Modeling RBI structure and behavior

In order to develop a language capable of explicitly describing the unique characteristics of RBIs, we began by identifying a set of high-level abstractions that capture the **structure and behavior** of TUIs (a common RBI style) [5]. Based on surveying existing TUIs, working with students in TUI classes, and building TUIs, we identified a set of high-level constructs for TUIs. We posit that the **structure** of a TUI is a set of relationships between two types of

physical objects: *tokens* which represent digital information (e.g. a building model in an urban planning TUI) and *constraints*, which constrain a token's manipulation (e.g. a surface on top of which a building model is manipulated). The relationship between a token and a set of constraints is called a *TAC (token and constraint)*. Similar to widgets, TAC objects encapsulate the set of manipulation actions that users can perform upon a physical object.

We have found that event models are not appropriate for describing a TUI's behavior. The **dynamic behavior** of a TUI is more perspicuously described using a model that captures both the linear aspect of a TUI (i.e. possible high-level states and transitions) and its distributed aspect (i.e. concurrent multiple users across multiple devices). Thus, we developed a two-tier model that combines elements from two modeling techniques: state machines and Petri Nets to describe the behavior of a TUI in terms of *states*, *tasks* and *actions.*

3. A Visual Language for RBIs

By identifying a set of core constructs and an interaction model we laid the foundations for a high-level description language for TUIs. We expect such a language to provide means to: 1) unambiguously specify the structure and behavior of a TUI 2) be comprehensible by non-programmers 3) explicitly capture the physical properties of interaction objects. To accomplish these, we are currently developing TUIML, a visual language for modeling and programming TUIs.

TUIML uses the concepts of tokens, constraints and TACs to describe both the **structure** of a TUI and the set of TAC **configurations** possible at runtime. TUIML provides a graphical notation that uses shapes to represent the physical form of tokens and constraints. This representation captures the physical characteristics of objects, thus implying how these objects can be manipulated in respect to each other. Figure 1 shows a simple TUIML description of tokens and constraints configuration in an urban planning application. To describe the **behavior** of a TUI, TUIML provides two types of diagrams corresponding to the two-tiers in our model.

We intend TUIML to be used in three ways: 1) as informal diagramming language for creating (mainly by collaborative sketching) diagrams which are aimed at exploring and understanding a solution space 2) as a semi-formal specification language for creating detailed and complete specifications which are aimed at guiding implementation activities. 3) as a visual programming language for creating complete and executable specifications.

We continuously evaluate TUIML while focusing on three aspects: expressiveness, usability and usefulness. Our evaluation activities include applying TUIML to describing variety of RBIs, and incorporating TUIML in a TUI class.

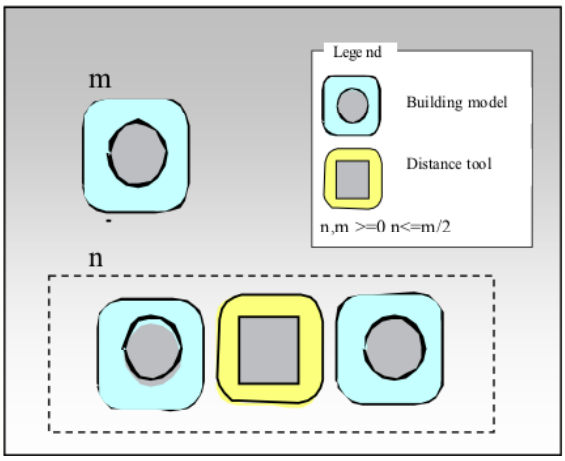

Figure 1. A TUIML description of the physical configuration of an urban planning application while performing the task of distance measuring. It shows *m* building models (tokens) that can be manipulated upon a surface (a constraint) while distance is being measured by connecting two buildings using a distance-measuring tool. Assuming n distance-measuring tools are available as well as sufficient building models, n users can perform the task of distance-measuring in parallel

4. Discussion and Future Work

We expect our research to contribute a technology-independent visual language that simplifies the development of RBIs. By allowing RBI developers from different disciplines to easily analyze, coherently discuss, and rapidly implement RBIs, we expect to lower the threshold for developing RBIs.

5. References

[1] B. Ullmer and H. Ishii, "Emerging Frameworks for Tangible User Interfaces", *IBM Systems* 39.

[2] M. Horn, R.J.K. Jacob, " Tangible Programming in the Classroom: A Practical Approach", *Extended Abstracts CHI 2006*.

[3] M. Nilsson, S. Johansson and M. Håkansson. "Nostalgia: An Evocative Tangible Interface for Elderly Users", *Extended Abstracts CHI 2003*.

[4] O. Shaer, N. Leland, E.H Calvillo and R.J.K. Jacob, "The TAC Paradigm: Specifying Tangible User Interfaces", *Personal and Ubiquitous Computing*, vol. 8, no. 5, pp. 359-369, Sept. 2004.

[5] O. Zuckerman, M. Resnick, “System Blocks: A Physical Interface for System Dynamics Simulation” *CHI 2003*.

[6] R.J.K Jacob, “What Is the Next Generation of Human Computer Interaction?”, CHI 2006 , workshop