TUIMS: Laying the Foundations for a Tangible User Interface Management System

Nancy Leland, Orit Shaer[1], Robert J.K. Jacob
Tufts University, Computer Science Department,
Medford, MA 02155, USA
{nleland, oshaer, jacob}@cs.tufts.edu

Abstract. This paper lays foundations for a Tangible User Interface Management System (TUIMS). It presents the TAC paradigm for representing TUIs, which identifies the core components of TUIs. Building upon this, it introduces TUIML, a high-level description language for TUIs. The concept of TUIMS is proposed and a built proof of concept prototype is discussed.

1 Introduction

Interaction with TUIs draws on users' existing skills of interaction with the real world [4], thereby offering the promise of interfaces that are quicker to learn and easier to use. However, these interfaces are currently more challenging to build than traditional user interfaces. In order to address the challenges unique to developing TUIs, we propose the concept of a Tangible User Interface Management System (TUIMS), drawing from earlier work on UIMS [2]. TUIMS allows designers to specify tangible interaction in a high-level description language (TUIDL). This specification would then be translated into a simulation program or a program controlling a set of physical interaction objects. The TAC Paradigm is a fundamental key in the development of TUIMS. It provides a unified conceptual framework for TUIs, thus reducing the different classifications of TUIs into a universal TUI class, making the development of a TUIDL possible.

2 The TAC Paradigm

The Token and Constraints (TAC) paradigm [3] aims to capture the formal structure of TUIs in the same way previous models have captured event-based interfaces. It was designed to provide a universal conceptual framework for TUIs that encompasses existing TUI classifications.

Our approach is based on describing a TUI as a set of relationships between two types of physical objects: **tokens** and **constraints**. *Tokens* are physical objects representing digital information or a computational function. Users interact with *tokens* in order to access or modify digital information. *Constraints* are physical objects that limit both the physical manipulation and the computational interpretation of a token. The relationship between a token and a set of constraints is called a **TAC**. Manipulation of a *TAC* is the manipulation of a *token* in respect to a set of *constraints* and it has computational interpretation. For example consider the Marble Answering Machine [1]. We consider the marble as a *token* and the replay indentation as its *constraint*. The pair marble and replay indentation is considered a *TAC*. The manipulation of a marble in respect to the replay indentation has computational interpretation: when the user places the marble in the indentation the message represented by the marble is played.

TAC components are the basic building blocks of TUIs just as widgets are for GUIs. To describe a TUI using the TAC paradigm framework, the TUI designer specifies a set of *TAC* objects and the actions performed upon them. These *TAC*s may be instantiated at run time by the user.

3 TUIML

Building upon the representation framework provided by the TAC paradigm we introduce TUIML (Tangible User Interface Markup Language). TUIML is a high-level description language for TUIs (TUIDL).

---

[1] and Design Machine Group, University of Washington, Seattle, WA, USA.

Drawing from the work on model based UIs [5], TUIML predefines five basic components: **Task, Domain, Representation, TAC** and **Control.** The *task* and *domain* components describe the semantics of the TUI. The *representation* and *TAC* components describe the syntax of the TUI system. The *representation* component defines a set of logical physical objects; the *TAC* component defines the context for interaction actions performed upon these logical physical objects and determines which semantic functions are invoked as a result of an interaction action. These components do not specify the TUI implementation mechanism. The *control* component keeps track of the TUI system state during run time and maps lexical level (implementation mechanism) events to the syntactic level interaction actions thus providing desirable technology independence.

4 TUI Management System

A TUIMS allows designers to specify a high level description of a TUI using TUIML. This specification would then be reused for generating a graphical simulation or implementing alternative prototypes. With a TUIMS an interactive application consists of two parts: a lexical handler handling the communication of the user with a set of physical objects and an application component containing the application logic. A lexical handler is provided for each implementation mechanism supported by the TUIMS. The TUIMS Dialogue Manager component is responsible for the communication between these two components.

We built a prototype TUIMS providing designers a 3d graphical modeling tool and form based tools to specify TUIs (see figure 1). The system translates the TUI description into a TUIML representation and simulates tangible interaction using a Java3D based VR lexical handler. We use this prototype to develop new TUIs.

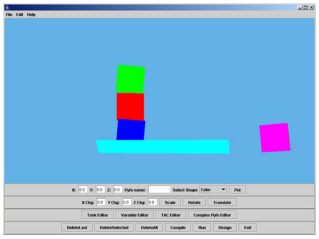

Figure 1, a TUIMS prototype provides graphical and form based tools to specify a TUI

5 Discussion and Future Work

We have presented the concept of a TUIMS and laid foundations for its development. The TAC paradigm identifies the core components of TUIs. Building upon this, we presented TUIML, a high-level description language for TUIs. Finally, we discussed the TUIMS concept and used it in building a prototype TUIMS.

We intend to develop a full TUIMS for specifying, programming and simulating TUIs. We are developing an automatic generator of interactive C code from TUIML specification supporting TUI prototyping using a Handyboard microcontroller. We are also interested in cooperating with existing physical toolkits to extend the technologies supported by the TUIMS.

6 References

1. Crampton Smith, G. “The Hand That Rocks the Cradle.” I.D., May/June 1995, pp. 60-65.
2. D. Olsen. “User Interface Management Systems: Models and Algorithms.” , Morgan Kaufmann, San Mateo, CA, 1992.
3. E.H. Calvillo Gamez, N. Leland, O. Shaer, and R.J.K. Jacob, "The TAC Paradigm: Unified Conceptual Framework to Represent Tangible User Interfaces," *CLIHC 2003 Latin American Conference on Human-Computer Interaction*, 2003.
4. Ishii, H., and Ullmer, B. “Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms”. In *Conference on Human Factors and Computing Systems*, March 1997.
5. P. Szekely. “Retrospective and challenges for model-based interface development.” In F. Bodart and J. Vanderdonckt, editors, *Computer Aided Design of User Interfaces (CADUI'96)*, pages 1–27, Wien, 1996. Springer-Verlag.