IdenTTop: A Flexible Platform for Exploring Identity-Enabled Surfaces

Grant A. Partridge
Dept. of Computer Science
University of Manitoba
Winnipeg, MB, Canada R3T 2N2
umpartr3@cs.umanitoba.ca

Pourang P. Irani
Dept. of Computer Science
University of Manitoba
Winnipeg, MB, Canada R3T 2N2
irani@cs.umanitoba.ca

CHI 2009, April 4–9, 2009, Boston, Massachusetts, USA.
ACM 978-1-60558-247-4/09/04.

Abstract
Only a subset of tabletop designs support the ability to determine which user has performed a given action. These *identity-enabled* (IE) surfaces offer significant functional advantages over systems with no such capability. Distinguishing between the two types of surfaces enables a valuable discourse that should serve to improve the usefulness of all tabletop designs. To facilitate examinations of the IE design space, we have developed a toolkit called IdenTTop, which greatly simplifies the process of prototyping new IE applications, and we present a few sample applications to demonstrate IdenTTop's effectiveness.

Keywords
Tabletops, multi-touch, identity-enabled surfaces

ACM Classification Keywords
H.5.2 [Information Interfaces and Presentation (e.g., HCI)]: Input devices and strategies (e.g., mouse, touchscreen); H.5.3 [Information Interfaces and Presentation (e.g., HCI)]: Group and Organization Interfaces---Collaborative Computing

Introduction
Tabletops are considered to support a sophisticated spectrum of interactivity. However, many common tabletop designs lack characteristics that interface developers take for granted in desktop environments.

One particular deficiency of popular multi-touch surface technologies like diffused illumination (DI) and frustrated total internal reflection (FTIR) is that they provide no inherent means of recognizing which user has performed a given action. Both of these methods use a computer vision system to detect blobs, which are then interpretable as gestures or taps. However, those actions cannot be attributed to a particular user. This significantly constrains the range of interaction that can be supported by applications on these tables.

Conversely, developers of applications for *identity-enabled* (IE) surfaces – like the DiamondTouch, for example – can take advantage of this capability to provide a richness of interaction unparalleled by simpler tabletop technologies. To understand the sorts of things that are possible in IE environments, consider the humble tools palette – a set of icons arranged in a grid, permitting quick visual confirmation of purpose and ready access at the click of a button, all with a minimum cost of display real estate. After changing tools, the effect of future clicks and gestures is modified appropriately. This widget has proved to be tremendously useful on the desktop; some commonly used applications with tools palettes include Microsoft Word, the Audacity audio editor, and practically every paint or graphics program ever popularized.

Palettes have traditionally been built to support a single user, but the concept can be easily extended into identity-enabled multi-user environments. Conventional, single-user palettes highlight the currently selected item. One approach might be to assign a color to each user and mark the selected item with that color. This would be suitable for identity-enabled displays that support a small number of simultaneous users; other strategies, for instance, rich embodiments with cursors [5], would probably be more appropriate in other environments.

Whatever the details of the implementation, one can easily adapt the tools palette to IE systems. This is not the case for applications on non-identity-enabled surfaces. Since user input cannot be differentiated, each user would be unable to select a tool independent of the other users' preferences.

The tools palette is just one example of an element that is easy to add to IE environments, yet resists simple implementation for non-IE systems. As a matter of fact, there is a whole class of interactive techniques possible on IE surfaces that cannot be accomplished easily without identity-enabling capabilities. Some authors have hinted at the true potential of identity-enabled systems, but the preliminary nature of research on this topic suggests that there might be a lack of enabling technologies to support the examination of this space in a convenient, flexible, yet powerful way.

Related Work

Almost all groupware implementations inherently afford the ability to distinguish users. For instance, identity differentiation is entirely natural in applications where each user has his or her own computer. The issue tends to arise primarily in single-display groupware (SDG) environments. One type of non-IE multi-user interface involves a desktop application in conjunction with a utility like CPNMouse [6], which allows the use of multiple USB mice with common Windows applications. Another similar alternative is MIDDesktop [4], which creates a multi-user Java environment populated with single-user applets. Since such programs or applets were not developed with multiple simultaneous users in mind, they are unable to recognize that there is more than one pointer, much less which belongs to a particular user. The limitations of such approaches are described more thoroughly by Hutterer et al [1]. Non-identity-enabled applications have proliferated only recently, with the introduction of surfaces intended for operation by multiple simultaneous users.

The most visible identity-enabled application development toolkit, DiamondSpin, demonstrates a significantly different approach [3]. DiamondSpin is undeniably powerful, but its sophisticated capabilities come at an expense to developers, inflating the cost of application development time. Furthermore, it is unsurprising that DiamondSpin is designed to take advantage of the power of the DiamondTouch tabletop in particular. On the other hand, there is a clear value in a system that aims for generality while still affording the sorts of interaction characterized by multi-user surfaces.

Ryall et al. [2] provide a survey covering the range of functionality permitted by IE surfaces. Their study is particularly important for framing the concept of identity differentiation. The topic of Ryall et al.'s paper is idWidgets, which are interface elements capable of determining the identity of the person interacting with them. However, by focusing their discussion on the widget level, they ignore the application and system level issues that arise. We argue that a top-down approach to these concepts can complement existing examinations of identity-enabled systems.

IdenTTop

To this end, we have created IdenTTop, a framework for prototyping and developing lightweight, identity-enabled multi-user applications. Programs built with IdenTTop can be made to work on any surface that operates as a Windows display, and any identity-enabled input mechanism can be supported through a very simple application programming interface (API). IdenTTop is developed in C# and uses graphics powered by DirectX.

To develop IdenTTop, we built our own custom IE tabletop that supports up to 4 concurrent users. It consists of a plain wooden table, a Polhemus electromagnetic motion tracker for input, and a ceiling-mounted projector for display.

IdenTTop's clean, modular architecture (see Fig. 1)

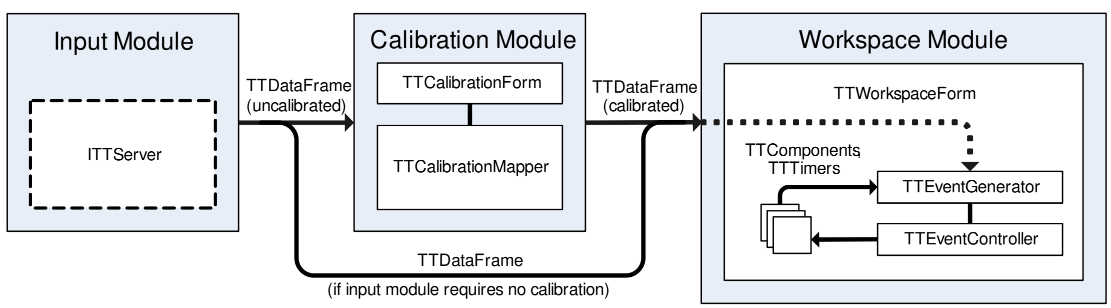

Figure 1. IdenTTop features a flexible and simple architecture.

consists of several useful components that make building tabletop applications almost as easy as single-user application development. *Input modules*, which are responsible for collecting raw user input, are built according to a simple interface, which makes it easy to support new input devices. IdenTTop is packaged with an input module that handles one or more USB mice, which makes it simple to develop multi-user applications on the desktop before trying the exact same code on a multi-touch surface. A *calibration module* can be inserted into the signal chain if the data from the input module requires further processing to yield meaningful coordinates. Finally, the *workspace module* allows developers to organize components within a workspace, which will be displayed upon the surface and will generate events based on input received.

The component system, used to implement widgets, is similarly flexible and useful. Besides the typical features one would expect, we have enhanced TTComponents with a few special capabilities geared towards the characteristics of tabletop interaction. For example, each TTComponent has an Orientation property which controls which edge of the component is the "top", rotating a widget's contents so that it may be comprehensible to every user, regardless of his or her position around the surface.

IdenTTop also supports drag-and-drop as an integral element of the toolkit. If a component's Draggable property is set to true, users may drag the component much like they would on a desktop. Event hooks are available to implement nonstandard dragging and dropping behaviors. Conversely, any component may be configured to act as a *drop receptacle* by implementing a function that receives the dropped component as a parameter and returns a value indicating the action to take upon drop. This provides a sophisticated context-sensitive means by which drags can be Accepted, Cancelled, or Deferred to a component further up the hierarchy.

Another useful characteristic of TTComponents is their hierarchical nature. Developers may designate any TTComponent as a container by flipping a flag, which enables the creation of nested interfaces of arbitrary complexity. The ability to nest components significantly reinforces the power of drag-and-drop; for example, these capabilities can be combined to automatically adjust the orientation of a component depending where it is dropped. This could be accomplished by defining several regions with large components.

As previously mentioned, DirectX handles IdenTTop's graphics. However, The canvasses of components abstract away the complexity of the DirectX API, instead presenting a GDI+ canvas that allows development of custom controls with a remarkably familiar interface (especially to seasoned .NET programmers). A special kind of component, the TTDirectXOverlay, can be added into the control hierarchy to provide a hook into the graphics loop which permits direct access to the lower-level graphics API and the power to incorporate hardware accelerated 3D graphics in a multi-user application.

Applications

To confirm the capabilities of IdenTTop, we have presented a few people with the toolkit with the goal of developing some applications for our tabletop. We now have several applications that demonstrate the value of the particular features emphasized in our design.

Stack-ollaborate

Our very first prototype was a multi-player cooperative game called Stack-ollaborate. In testament to the simplicity of IdenTTop, a single developer built Stack-ollaborate in an afternoon. In this game, up to four players race together against the clock to stack tiles in a few piles by dragging and dropping them as quickly as possible, while coordinating the tiles by suit and rank. IE capability is used to track the activity of individual players, so that this game might be useful as a test-bed for measuring successful collaboration.

Drum Circle

Drum Circle is a collaborative musical toy that was essentially developed in two hours. It demonstrates IdenTTop's ability to orient components to face any edge of the screen, as well as its hierarchical component structure. Drop receptacles are grouped in "measures" of four beats and arranged around the table. Four players drag and drop blocks representing different drum sounds into the receptacles. A flashing light travels around the table from beat to beat, triggering the sound in each bin and forming a rhythm that can be sped up or slowed down with a few controls in the center.

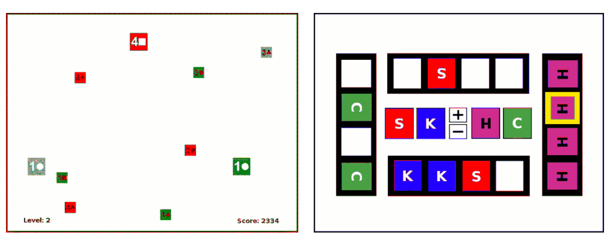

Figure 2. Stack-ollaborate (left) and Drum Circle (right).

WallBalls

WallBalls is an unconventional musical instrument designed for simultaneous play by up to four people at once. A TTDirectXOverlay allows players to draw walls onto a grid. The color of each wall indicates the sound that will be triggered when a ball is thrown at it by dragging and flicking. Balls freely ricochet around the table, triggering sounds in a more or less unpredictable manner. MIDI messages are sent to an external module for sound production. The design of WallBalls is extensively reliant upon the identity-enabled capabilities of IdenTTop. Besides the two tools palettes, there are also linked, replicated widgets that allow any player to temporarily seize exclusive control of a particular parameter (in this case, the speed/direction of all balls on the table), and a group of linked buttons which clear the table only when pressed simultaneously by a majority of players. WallBalls is a much more sophisticated application than the other two apps described, and it serves to exhibit the robustness and viability of the IdenTTop platform as an solution for more complicated projects, even at this early stage.

Future Work

We have identified a major gulf of functionality between IE and non-IE surfaces. Acknowledging this division raises many important questions governing the design of tabletop applications:

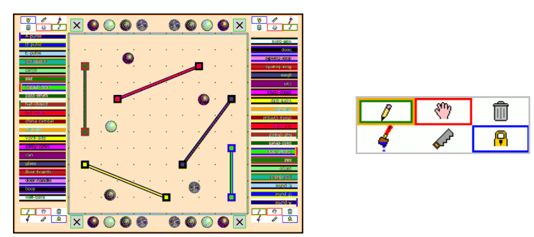

Figure 3. WallBalls and its multi-user tools palette.

- What can we accomplish with IE tables that would be much more difficult or impossible to achieve with non-IE systems?
- What sorts of characteristics are common to identity-enabled applications? How about non-IE apps?
- How could applications built for non-IE environments be served better by an identity-enabled surface?
- Are there any viable strategies for simulating identity recognition in systems that offer no native support?

These questions and more provide a wealth of opportunity for exploration. By separating these systems into identity-enabled and non-IE categories, we hope to draw attention to the significant disparity of interactive power between the two types. Even preliminary consideration of these reveals interesting insights. For example, a gesture-based strategy might be able to capture some of the advantages of IE systems in a non-IE environment if each user was to identify him or herself with a unique gesture.

IdenTTop's workspace module is an ideal platform to host a library of identity-enabled widgets like idWidgets [2]. However, development of such a collection raises lots of questions about the most effective way certain tasks should be accomplished in the multi-user space. It stands to reason that building a widget library before knowing which ones are actually useful is a dangerous practice. As a result, we are approaching this task very carefully. We intend to experimentally compare a number of widget designs inspired by current work in the field. Some of these will intrinsically take advantage of IE capabilities, while others will be compatible with non-IE approaches. In this way, we can compare the effectiveness of widgets supported by each type of table, and by extension, gain some insight into the advantages that IE tables might have over non-IE systems.

Acknowledgements
We acknowledge the financial support of the National Sciences and Engineering Research Council (NSERC). We would especially like to thank Jonathan Cullen, David McCallum, Mat Tate and Sean Gustafson for assistance with IdenTTop's development.

References
[1] Hutterer, P. and Thomas, B.H. Groupware support in the windowing system. In *Proc. 8th Australasian User Int.*, 39-46, 2007.

[2] Ryall, K., Esenther, A., Forlines, C., Shen, C., Shipman, S., Morris, M.R., Everitt, K., and Vernier, F. Identity-differentiating widgets for multiuser interactive surfaces. *IEEE Comp. Graph. Appl. 26*,5 (2006), 56-64.

[3] Shen, C., Vernier, F.D., Forlines, C., Ringel, M. DiamondSpin: an extensible toolkit for around-the-table interaction. In *Proc. CHI,* 2004, 167-174.

[4] Shoemaker, G.B.D., and Inkpen, K.M. MIDDesktop: an application framework for single display groupware investigations. Technical Report TR 2000-1, Simon Fraser University, 2000.

[5] Stach, T., Gutwin, C., Pinelle, D., and Irani, P. Improving recognition and characterization in groupware with rich embodiments. In *Proc. CHI*, 2007, 11-20.

[6] Westergaard, M. Supporting multiple pointing devices in Microsoft Windows. In *Proc. MS Summer Workshop for Faculty and PhDs.*, Cambridge, 2002.