Proceedings of the World Conference on Innovative VR 2009
WINVR09
July 12-16, 2008, Brussels, Belgium

WINVR09-740

DRAFT: SPARSH UI: A MULTI-TOUCH FRAMEWORK FOR COLLABORATION AND MODULAR GESTURE RECOGNITION

Prasad Ramanahally
Department of Computer Engineering

Stephen Gilbert
Human Computer Interaction, Department of Psychology

Thomas Niedzielski
NSF REU Intern

Desirée Velázquez
NSF REU Intern

Cole Anagnost
NSF REU Intern

Virtual Reality Applications Center
Iowa State University, Ames , Iowa – 50011, USA

ABSTRACT
Most current multi-touch libraries provide support to recognize the touch input from particular hardware and seldom support complex gestures. For rapid prototyping and development of multi-touch applications, particularly for collaboration across multiple disparate devices, there is a need for a framework which can support an array of multi-touch hardware, provide gesture processing, be cross platform compatible, and allow applications to be developed in the desired programming language. In this paper we present criteria for evaluating a multi-touch library and “Sparsh UI”– an open source multi-touch library which is a novel attempt to address these issues by enabling developers to easily develop multi-touch applications. We also compare Sparsh UI with other multi-touch libraries and describe several Sparsh-based applications, including BasePlate, a system for collaborative virtual assembly.

Keywords: Multi-Touch, Collaboration, Gestures, Prototyping

1 INTRODUCTION
Multi-touch is a human-computer interaction technique that allows users to interact with a system without the conventional input devices, such as a mouse or keyboard. Typical multi-touch systems consist of a touch screen (table, wall, etc.) or touchpad, as well as software and hardware that can recognize multiple simultaneous touch points. Standard touch screens, such as computer touch pads or ATM machines, generally recognize only one touch point at a time.

Multi-touch is a growing area of research with a variety of both hardware input devices and software libraries for handling multi-touch input. Each of these software systems could be considered a competitor for a common standard for multi-touch applications, something which has not yet emerged. This research presents a framework for comparing multi-touch software protocols and one particular approach, Sparsh UI, an open source multi-touch gesture recognition library that is device agnostic and is capable of running on a variety of platforms. Several sample applications that use Sparsh are described.

Collaboration among different multi-touch hardware devices necessitates the application to be compliant with multiple hardware and platforms. The application should also be capable of processing gestures for a variety of applications and interaction designs.

2 DESIGN CRITERIA
A generic multi-touch gesture recognition framework can be evaluated based on its approach to addressing the following challenges:

1) The support of a variety of multi-touch input devices.
2) Gesture recognition.
3) Support of different platforms e.g. Windows, Linux and Mac.

4) Support of different development languages e.g. Java, C++.
5) Interface scale, e.g., finger input vs. whole hand input
6) Simultaneous collaboration of multiple users

Each component of this framework will be described in more detail below, followed by a description of how Sparsh UI addresses that challenge.

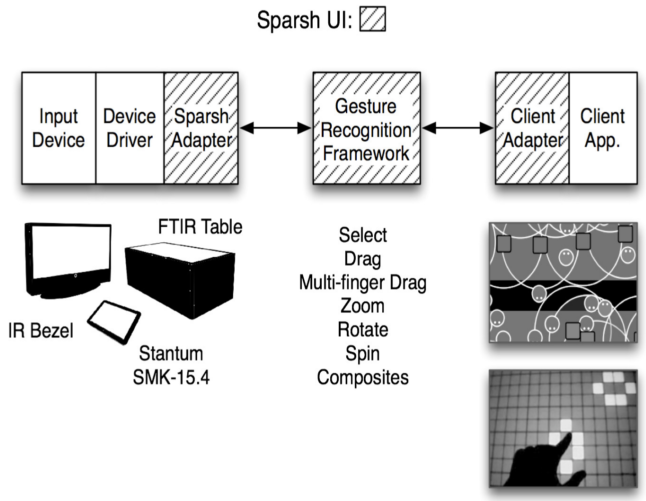

Figure 1: The Sparsh UI Architecture: The Sparsh Adapter standardizes touch events from varied hardware, sends the events over TCP to the Gesture Recognition Framework, which then sends appropriate events to the software client via the Client Adapter.

2.1 Ability to a Support Variety of Hardware.

In order to support a variety of hardware devices, a multi-touch library must be able to standardize the input data format from the different devices.

Most multi-touch devices generate the following information from a touch, in addition to the (x, y) coordinate values on the screen:

1) Information related to the creation of the touch point when the finger comes in contact with the screen (henceforth referred to as Point Birth).
2) Information related to the motion of the touch point (referred to as Point Move).
3) Information related to release of the finger from the touch screen (Point Death).
4) An identification number for each touch point generated when a finger comes in contact with the touch screen (Point ID).

Thus each touch point can be distinctly identified by the x-y coordinates, the state of the touch point (Point Birth, Point Move and Point Death) and the ID of the touch point (Point ID). Sparsh UI specifies a format in which a device should send touch data to it and provides a driver adapter that can standardize the input from the driver which would be compatible with Sparsh UI (Figure 1).The driver adapter uses the data structure with the parameters mentioned above. As of now we have Sparsh compatible drivers for a 60" FTIR-based touch table created at Iowa State University, a Stantum SMK-15.4" multi touch tablet, and a 42" infrared-based bezel display from IR Touch. An adapter for the Dell Latitude XT is in progress.

Communication between the device driver and the Sparsh UI gesture recognition framework takes place over sockets using TCP protocol. This allows the driver and driver adapter to be written in the language of choice.

2.2 Gesture Recognition

Most contemporary open source multi-touch software libraries provide only the ability to recognize the touch points and pass the touch coordinates directly to the application, leaving the application to do the gesture processing.

Multi-touch is made intuitive by means of gestures and it is vital for a multi-touch library to provide gesture recognition support. The following considerations should be addressed while providing gesture support to multi-touch applications:

- Flexibility to specify the supported gestures at an application level and UI component level.
- Support for providing touches point coordinates if the application does need to do custom gesture recognition.
- Ease of adding new gestures to the framework.

The usage of various gestures can be specific to the application. For example, our image manipulation application Picture App makes use of the Drag, Zoom and Rotate gestures, but another application called "Table Tower Defense" just makes use of touch gestures. One needs to process the touch point data to recognize all possible gestures for the former and just send out the touch coordinate data for the latter. Hence it is inefficient to analyze raw touch data for various gestures unless the application needs it. Similarly, not all the UI components would require all the gestures. For instance, a button would allow only Select, while a window title bar would allow Drag and other gestures that indicate minimize, maximize, etc. To incorporate this flexibility, we use the concept of Group ID to identify the various UI components on the screen. On each point birth, the application is queried for a Group ID and the allowed gestures corresponding to the point location. Different point birth sequences can be associated to the same Group ID (analogous to multiple fingers on the same UI component).If no gesture recognition is required for a given touch coordinate, it can return a null value indicating that there is no need for gesture processing. The Sparsh UI gesture recognition framework processes the incoming touch point coordinate data, recognizes the associated gestures, and sends back the

associated gesture event for the Group ID. At the application end, it can be easily identified as to which component has been acted upon and the gesture event can be handled appropriately.
Applications can register for receiving the raw touch points if they need to process special custom gestures. The modular design of Sparsh UI makes it easy to add new gestures to the gesture recognition framework. Currently we are working on dynamically loading gesture modules
The intuitiveness of a multi-touch interface is achieved through the use of gestures which are highly intuitive in the application context in which they are used. The usefulness of a gesture library would in part depend upon the number of intuitive gestures that a library can support. Sparsh UI currently supports the following gestures. More gestures are being added as a part of continuous improvement of the framework.

2.2.1 Select Gesture Simply placing a finger on the multi-touch device performs this gesture. This gesture is normally used for selection purposes. Although it can also be used creatively as in our Table Tower Defense game. The touch coordinates are passed to the application whenever the gesture framework detects this gesture.

2.2.2 Spin Gesture Spin is the newest addition to the Sparsh UI gesture list (Figure 2). This gesture is performed by placing two fingers on the multi-touch device that creates an invisible axis, somewhat similar to Jeff Han's two-handed hold-and-tilt gesture [1].In a 3D CAD-like application, once the axis has been established by one hand, the user is able to spin the 3D view point by dragging a third finger perpendicular to the axis created by the first two fingers. It can be used for any chosen axis of rotation. This gesture can be used to manipulate views in any 2D or 3D environment.

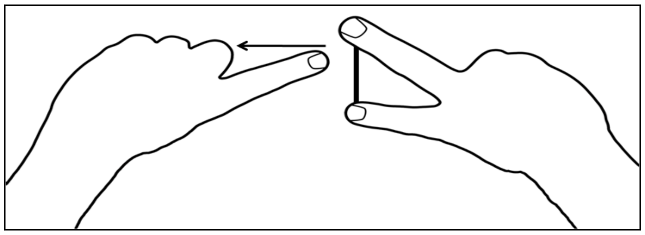
Figure 2: Example of spin gesture.

2.2.3 MULTI-FINGER DRAG GESTURE The multi-finger drag gesture is a generic drag gesture which detects the drag (or swipe) when one or more fingers are moved across the touch screen. If applications need not differentiate between the number of fingers that are used to do the drag operation, they can register for the multi-finger drag gesture.
In all the drag gesture implementations, the gesture recognition framework generates drag events with the parameters ΔX and ΔY, the amount of offset from the initial position (initial position of centroid if it's not single touch).

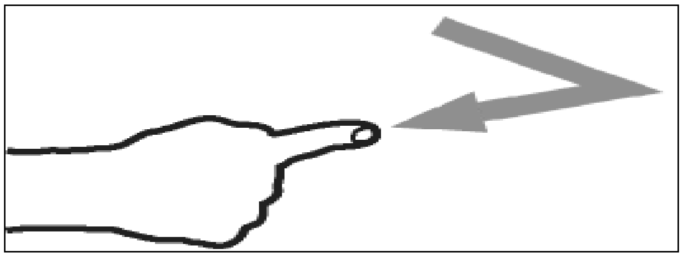
Figure 3: Example of one-finger drag gesture.

If an application needs to distinguish between the number of fingers that resulted in the drag gesture, it can register for one or more of the following gestures:
One-finger drag: The user performs this gesture by placing a finger on the device and dragging it across the surface (Figure 3). This gesture can be used for moving graphic elements on the screen. It can also be used for panning a view (e.g., panning a map).
Two-finger drag: This is similar to one-finger drag gesture except that two fingers are used instead of one. The two fingers may be held close to one another or apart.
Three- finger drag: In this case three fingers are used to perform the drag or swipe operation. This gesture is being considered to manipulate 3D objects in a multi-touch environment where both 3D and 2D objects are present. Similarly, Sparsh UI offers gestures for four-or five-finger gestures.

2.2.4 Rotate Gesture This gesture is performed by placing two fingers, either from the same hand or different hands, on the multi-touch device and rotating them clockwise or counter-clockwise (Figure 4). The gesture framework generates an event with the parameters consisting of the angle of rotation and the coordinates of center about which rotation occurs.

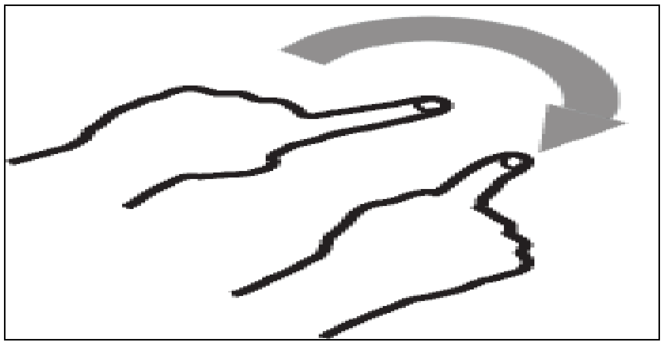
Figure 4: Example of rotate gesture. Both fingers may also move to perform the rotate.

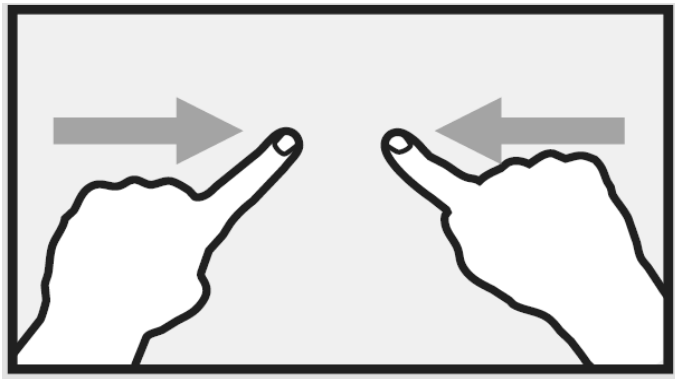

Figure 5: Example of zoom gesture, in this case zooming out.

2.2.5 Zoom Gesture This gesture (Figure 5) is performed by placing two fingers on the multi-touch device and dragging them away or towards each other along a line. The gesture is typically used for zooming in and out of maps, or more generally, scaling and resizing screen elements. When a zoom gesture occurs, an event consisting of the scale factor is generated.

2.3 Support of Different Platforms

A generic framework should be capable of operation across popular operating systems (Windows, Linux, and MacOS X). Sparsh UI exists in both Java, which is supported my most popular operating systems, and in C++ version using the Boost library, which makes it cross platform compatible.

2.4 Support of Different Languages

Since Sparsh UI uses socket based communication to communicate with the multi-touch application, this allows one to write Sparsh-based applications in language of choice. Currently we have client adapters for Java / C++ which abstract the communication protocol over sockets between the client application and the gesture framework. In future we plan to provide client adapters for other popular programming languages.

2.5 Support of Wide Interface Scale

Since collaboration across multi-touch devices often use disparate multi-touch devices of varying dimensions, it is important that gesture processing is not affected by the varying resolutions of different devices. This is achieved by using relative values for touch coordinates instead of absolute coordinates. However it is the responsibility of application developers to ensure the usability and ergonomics of applications across devices of varying dimensions. We are planning to provide support for getting information regarding physical dimensions of the device, resolution, etc. in Sparsh UI.

2.6 Support of Collaboration

Sparsh UI provides a platform to develop collaborative multi-touch applications where collaboration is achieved at application level, e.g., by using TCP sockets to have two instances of the same application on different systems exchange data. In future versions we plan to support collaboration in Sparsh-UI so that various gesture events can be exchanged across networked multi-touch devices. But nevertheless there will always be application data which needs to be exchanged at application level.

2.7 Summary of Framework

Table 1 compares Sparsh UI with TouchLib by the NUIGroup [6] for the various features discussed so far.

Feature	Sparsh UI	Touch Lib
Multi-Hardware compatibility	Yes	No
Gesture Recognition	Yes	No
Cross Platform support (Linux/Windows/Mac)	Yes	Yes
Multi-language support	Yes	No
Interface scale	Pending	No
Direct collaboration support	Pending	No

Table 1: Comparison of Sparsh UI and other multi-touch libraries.

2.8 Ease of writing Multi-touch applications

Sparsh UI eases the process of writing a multi-touch application and enables rapid prototyping of multi-touch applications. Since Sparsh UI takes care of hardware abstraction and gesture recognition, the developer's task is greatly simplified. The framework provides helper libraries application. As a part of Sparsh UI, framework helper libraries (for C++ and Java currently) are provided to abstract all the communication with the Sparsh Gesture Framework. To develop a multi-touch application with Sparsh UI the following needs to be done:

1) Have a mechanism to uniquely identify each UI component on the screen by means of a group ID. This would enable the application to immediately act upon the UI component when an event is delivered for a particular group ID.
2) Write event handlers for the various gesture events delivered by the Sparsh UI framework. Whenever a gesture event occurs, a call back function is called where one would need to identify the type of gesture event and call the appropriate event handler.

The above process is much simpler than writing custom gesture recognition code.

2.9 Touch Simulator

As a part of our framework in order to test the multi-touch applications, we have a touch simulator which can be used to simulate multi-touch input using mouse input. Multi-touch inputs can be simulated by freezing the screen (pressing "Esc" triggers the freeze in our simulator) and using the mouse clicks and drags to simulate the touch input of fingers. When a developer has accumulated all the simulated touches needed, she can exit out of "freeze mode," and all the clicks are given as simultaneous touch inputs to the gesture recognition framework. The touch simulator also takes care of assigning proper IDs and states to the simulated touch points. This touch simulator expedited the testing of various multi-touch applications that we developed using Sparsh UI. Figure 6 shows a snapshot of touch simulator in the freeze mode when the multi-touch events are recorded. The various tracks shown in the figure indicate the path traversed by mouse pointer, the circular spots mark the Point Birth and Point Death which are simulated by button click and button release respectively.
On exiting from the freeze mode, the simulated events are activated.

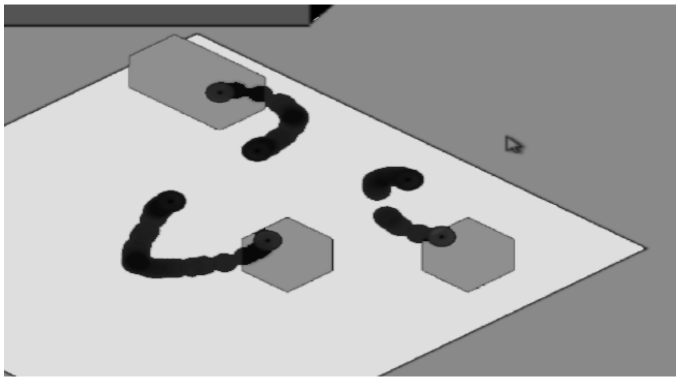
Figure 6: Example of drag sture in the Touch Simulator, in this case dragging objects in a circular path.

3 IMPLEMENTATION EXAMPLES USING SPARSH UI

3.1 Remote Collaboration in Multi-Touch Environment for Computer Aided Design - BasePlate.

A multi-touch environment, in conjunction with the use of intuitive gestures, can offer an excellent platform for design CAD tools of the future. Previous works [7] have shown how collaboration can contribute to better understanding and efficient solving of a problem. Hence allowing distant users to collaborate on a CAD project in a multi-touch environment may enable faster design work.
To explore this concept we devised a simple collaborative multi-touch application called BasePlate. This multi-touch application is inspired by LEGO® bricks (Lego). Users can build structures by placing 3-dimensional blocks on a plate. The application is collaborative in nature, e.g., users can see what other participants are doing. If user A moves a block by dragging it, user B sees it move also. The common structure is displayed to all users who are participating in the shared task. However, each user can have his or her own individual view (orthogonal or perspective). Also, one can identify which blocks are arranged by which user. The touch gesture is used to select and place a block on the BasePlate, the drag gesture to move the block on the BasePlate, and the spin gesture to rotate the view of the BasePlate. Figure 7 shows a screenshot of the application.

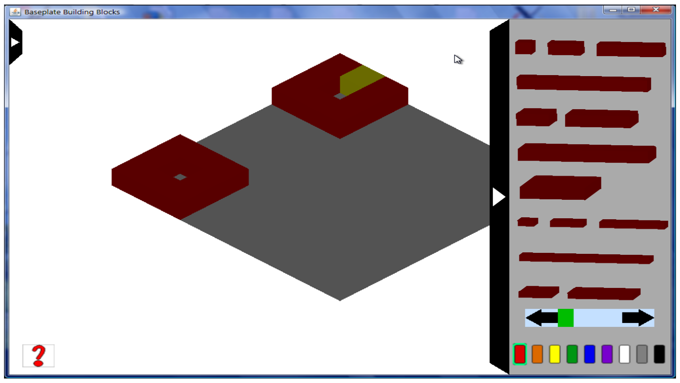

Figure 7: Screenshot of BasePlate – a collaborative multi-touch application.

3.2 Table Tower Defense

Table Tower Defense is a game that was developed keeping in mind that two to six people can simultaneously participate on a large multi-touch device like the Iowa State University 60" FTIR-based Touch Table. This game is a good display of the power of collaboration (on the same device) that can be achieved using multi-touch devices. It's a simple game in which participants on one side send tiny "creeps" or missiles to the opposite side and defend their own territory by building towers and destroying the creeps sent by opponents

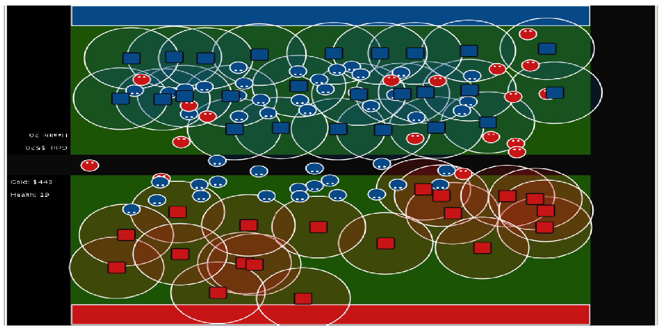

Figure 8: Screenshot of Table Tower Defense – a multi-user, multi-touch game.

This collaborative game, in which multiple users are using natural touch-based gestures to interact directly with the game elements they care about, would not exist without the use of multi-touch. Figure 8 shows a screen shot of the game and the video of game in action can be found at [5].

3.3 Picture Application

This is a simple photo application where one can use gestures to manipulate photos. Traditional gestures of Select, Drag, and Zoom are used.

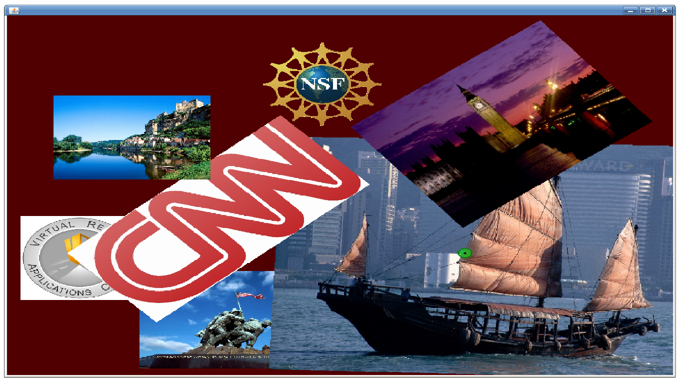

Figure 9: Screenshot of the Picture App

4 CONCLUSION

We have described criteria for evaluating a multi-touch software library, presented Sparsh UI and how it compares with TouchLib. We also presented the helper tools available as a part of our framework that expedite the development and prototyping of multi-touch applications. We also presented a novel approach of utilizing collaboration in virtual assembly environment using multi-touch. Real-time collaboration will likely become more important to CAD in the future and is worth exploring with new applications such as BasePlate.

5 FUTURE WORK

Currently we are working on adding more gestures to the gesture framework. We are also working on resolving conflicts when multiple gestures occur in conjunction and on resolving scenarios where disparate actions from separate users can conflict, appearing to be a single gesture. Future applications include more complex virtual assembly for manufacture and military command and control applications in which users collaborate across environments, e.g., from inside the Iowa State University VRAC's C6 and C4.

6 ACKNOWLEDGEMENTS

We thank students who participated in this research: Satyadev Nandakumar, Rob Evans, Anthony Ross, Jay Roltgen and Peter Wong. Research on Sparsh UI was sponsored in part by the Grow Iowa Values Fund and part by the US Air Force Research Lab. Work on BasePlate was performed at Iowa State University as part of a research internship sponsored by NSF (IIS-0552522), the Human Computer Interaction Graduate Program, and the Program for Women in Science and Engineering.

7 REFERENCES

[1] Dohse, K.C.; Dohse, T.; Still, J.D.; Parkhurst, D.J., "Enhancing Multi-user Interaction with Multi-touch Tabletop Displays Using Hand Tracking," *Advances in Computer-Human Interaction, 2008 First International Conference on* , vol., no., pp.297-302, 10-15 Feb. 2008

[2] Grossman, T.; Wigdor, D., "Going Deeper: a Taxonomy of 3D on the Tabletop," *Horizontal Interactive Human-Computer Systems, 2007. TABLETOP '07. Second Annual IEEE International Workshop on* , vol., no., pp.137-144, 10-12 Oct. 2007

[3] Leganchuk, A.; Zhai, S.; Buxton, W., "Manual and Cognitive Benefits of Two-Handed Input: An Experimental Study,"*Trans. on HCI 5(4)*, vol., no., pp. 326–359, Dec. 1998.

[4] Han, J. (2006, August). *Jeff Han demos his breakthrough touchscreen |Video on TED.com.* Retrieved July 24, 2008, from TED: Ideas worth sharing:http://www.ted.com/index.php/talks/jeff_han_demos_his_breakthrough_touchscreen.html

[5] http://www.youtube.com/watch?v=71BfgXZVBzM

[6] Natural User Interface Group : Open Source Multitouch - http://nuigroup.com/wiki/touchlib/

[7] Arias, E., Eden, H., Fischer, G., Gorman, A., and Scharff, E. 2000. Transcending the individual human mind—creating shared understanding through collaborative design. ACM Trans. Comput.-Hum. Interact. 7, 1 (Mar. 2000), 84-113