Capturing and Indexing Computer-based Activities With Virtual Network Computing

Sheng Feng Li
Department of Engineering[♣]
University of Cambridge
Cambridge CB2 1PZ, UK
+44 1223 766513
sfl20@eng.cam.ac.uk

Mark Spiteri, John Bates
Department of Engineering
University of Cambridge
Cambridge CB2 1PZ, UK
+44 1223 766516
{mds24, jb141}@cam.ac.uk

Andy Hopper[♣♠]
AT&T Laboratories Cambridge[♠]
24a Trumpington Street
Cambridge CB2 1QA, UK
+44 1223 343000
ah@uk.research.att.com

ABSTRACT

In this paper, we present a new technique to capture and index computer-based activities, without hindering natural human-computer interactions. This technique is based on the Virtual Network Computing (VNC) technology, which is an ultra-thin-client/server computing model that separates the display interface from the application logic in windowing systems. The server executes all the applications and the client simply presents the frame buffer updates to the user and accepts user input. We record the frame buffer updates for work review, and store the user and system events as potential indices into the recording.

Keywords

VNC, capturing, indexing, multimedia, events

1. INTRODUCTION

In much of the work we do, our activities tend to be forgotten once our goal has been achieved. These activities which could potentially document exactly how we reached the goal are then no longer referable. However, we may be required to perform similar or related work in the future, so the knowledge and experience we gained in the past could be of assistance. Capturing our activities therefore provides a source of records that can be revisited as effective resources for further work, while indexing these records enables their access to be random and hence more efficient. Many computational tools collect and analyse audio, video and computing recordings of human interaction to support the real-time capture of and subsequent access to users' activities [2]. Although these generic tools can also be applied to capture and access computer-based activities, human-computer interaction provides a richer set of indices that we can extract from its recording. For instance, every user event such as a keystroke in a computer-based work session can be a potential index into the session's recording. In this paper, we present a new technique to capture and index computer-based activities with Virtual Network Computing (VNC) developed at the AT&T Laboratories Cambridge [4].

2. VIRTUAL NETWORK COMPUTING

The new trend of client/server computing is to move as much computation and state information from the client to the server as possible to keep the client thin. VNC pushes this thin-client computing model to an extreme by moving the execution of applications completely to the server, leaving the client stateless. Without changing any windowing system, VNC runs on its platform and breaks it into client/server pieces, namely the VNC server and the VNC client. Figure 1 shows the VNC client (also known as the VNC viewer) running as a Java applet in the Web browser.

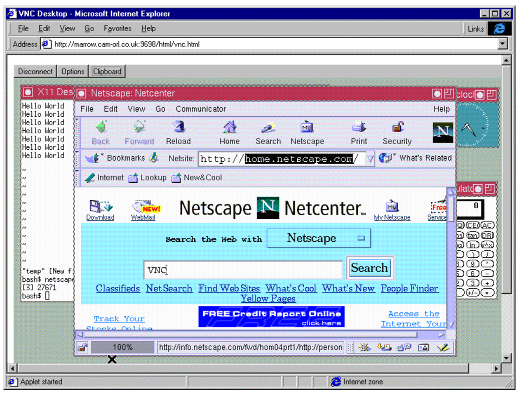
VNC Desktop - Microsoft Internet Explorer
File Edit View Go Favorites Help
Address http://marrow.cam-orl.co.uk:9698/html/vnc.html
Disconnect Options Clipboard
X11 Des
Netscape: Netcenter
Back Forward Reload Home Search Netscape Print Security
Netsite: http://home.netscape.com/
Netscape Netcenter
Search the Web with Netscape
VNC
Search
Applet started
Internet zone

Figure 1. VNC viewer operating netscape, xterm, xcalc, and xclock

The VNC client sends the following messages to the server:

- *Frame buffer update request* notifies the server that the client is ready to receive and display the next frame buffer update.
- *Key event* indicates a key press or release when the client receives key input from the user.
- *Pointer event* indicates either pointer movement or a pointer button press or release when the client receives mouse input from the user.
- *Client cut text* shows that the client has new ASCII text in its cut buffer. This facility was originally designed for cutting from a local session and pasting to a VNC session, but users can also use the facility to type a note in the client's clipboard and send the textual annotation across to the server. Figure 1 shows a push button labelled "Clipboard" at the top of the VNC viewer applet. When this button is

pushed, a clipboard window will appear, inviting users to take a note about the activity.

The VNC server sends the following messages to the client:

- *Frame buffer update* replies to the *frame buffer update request* message from the client. This message consists of a sequence of rectangles, each representing a rectangular area of the frame buffer. Together, these rectangles cover the frame buffer area which has been changed since the last update. Each rectangle uniquely identifies a rectangular area of the frame buffer by specifying the *x, y coordinates* of the area's upper-left corner, and its *width* and *height*. The rectangle also specifies the encoding scheme the pixel data of its represented area will follow. In the case of raw coding (i.e. no encoding at all), it lists the pixel values in the scan-line order, i.e. from left to right and from top to bottom.
- *Bell* rings a bell on the client to notify the user that something has occurred, e.g. when new emails have arrived in the mailbox.
- *Server cut text* shows that the server has new ASCII text in its cut buffer. This facility was originally designed for cutting from a VNC session and pasting to the local session, but can be used to mark some text on the screen as a point of interest. The server generates the ASCII text and sends the plain text across to the client when the user highlights a text string from the frame buffer. As shown in Figure 1, the server will send "home.netscape.com" to the client, as this string is highlighted in the HTTP address field of the Netscape Navigator in the VNC viewer.

3. COMPUTER-BASED ACTIVITIES

We seamlessly introduce a proxy server between the VNC client and server to intercept and manipulate the message streams in the VNC session. The proxy server timestamps the messages and saves them to some permanent storage. The timestamps serve as synchronisation points between message sequences. The *frame buffer update* messages are stored in a log file. The log file alone is sufficient for the playback of the activities captured in a VNC session. An asynchronous VNC client (i.e. the VNC reviewer) can read and display the *update* messages from the log file in the same way that the synchronous VNC client (i.e. the VNC viewer) does from the VNC server. The resulting movie allows the user to review the captured activities with VCR-like control. However, the benefits of the reviewing are often overshadowed by the tedious sequential access to the recording. To provide the user with browsing and searching capabilities, the messages such as the *key events* can be combined to provide indexing points. These messages, which we call *event* messages, are saved in the event store [5]. With the help of their timestamps, the *event* messages can be mapped to the synchronisation points in the log file of *frame buffer update* messages to allow random access of the recording.

A *frame buffer update* message usually affects only a small area of the frame buffer, which makes the access to these captured messages strictly sequential. To enable random access, we insert optional checkpoints to the log file of *update* messages at certain intervals. A checkpoint is an *update* message that contains the pixel data for the entire frame buffer. While searching for indices, the VNC reviewer reads in the nearest checkpoint and retrieves a full screen image, then it goes through the log file from there with a number of screen updates until the requested screen is found.

3.1 From Events to Indices

Minneman et al. explored a variety of methods for creating indices into captured multimedia records and considered the indices in four broad classes [2]. We identify *event* messages as indexing points into the captured *frame buffer update* messages, and we shall now explain how we convert these *events* into indices using the categories Minneman identified. Firstly, there are *intentional annotations*, which are indices that users create during a work session for the purpose of marking particular time points or segments of activities or points of interest. Our VNC client allows the user to take brief notes about the computer-based activities as they progress, and the timestamp of a note (i.e. a *client cut text* message) makes it an index into the recording. Another example of *intentional annotations* is that a user can highlight a text string of interest on the screen and save the marked text as a *server cut text* message. Secondly, there are *side-effect* indices, which are activities whose primary purpose is not indexing. These activities provide indices because they are automatically time-stamped and logged. Examples of such indices include *key event, pointer event* and *bell*. The purpose of *key event* and *pointer event* is for the user to interact with the computer and the purpose of *bell* is to notify the user that something has happened. But these events may be what a user remembers when she plays back a recorded work session and wishes to review the part where a particular keyword was typed or when the bell rang as the result of email arrival. The facilities provided by the *client cut text* and the *server cut text* may still be used for their original purposes, i.e. for cut-and-paste, and the text being copied across sessions provides *side-effect* indices too. Thirdly, there are *derived indices*, which are produced by automated analysis of detailed multimedia records. For example, by analysing the *frame buffer update* messages, we can calculate the size of the frame buffer area that has been changed since the last update to make screen change indices. A big screen change may suggest that an application window has been opened or closed, and thus is a potential useful index into the overall activity. Finally, there are *post hoc indices*, produced by anyone who later accesses the activity records. Our VNC reviewer allows the user to annotate the recorded VNC session during the review process. Like *intentional annotations*, these *post hoc* annotations carry timestamps and can be collected as indices.

3.2 Replaying, Browsing and Searching

The VNC reviewer replays the captured activities similar to playing back a video tape. However, the resulting movie suffers from the time-consuming serial access, and our indexing facilities play an important role to enable random access.

Our reviewer provides various kinds of browsers with different interfaces to allow users to browse through the captured activities. One such browser is called the thumbnail browser. When the user enters a number, say 40, in the "update" text field, the browser translates the number into an SQL query:

```
select * from frame-buffer-update where update >= 40%;
```

which means:

"Select all the frame buffer update events that change 40% or more of the screen area"

This query is then sent to the event store and all the events that meet the conditions are retrieved. Using the event's timestamp, we can locate its corresponding synchronisation point in the log file of *frame buffer update* messages. The screen image at that location is converted into a thumbnail image and displayed in the browser. Selecting one of the thumbnail images will cue the reviewer to begin playing the activities from that time point.

Our reviewer also provides the searching facilities for users who wish to search for a particular event or event sequence. Here we explain how the searching works with an example:

- *Key event*

When the user selects the *key event* option and enters the word "ls" in the "search" text field, the reviewer composes the following SQL query:

> select * from key-event where key = 'l' followed-by key = 's';

which means:

> "Select all the *key event* sequences where the first key press is the letter 'l' immediately followed by a second key press 's'"

Similar to the "Find" facility in Microsoft Word, the user can specify whether the search should "match case" or "find whole words only".

The translation of search items into SQL queries is hidden from the user. However, she can choose to display and modify these queries before they are sent to the event store. All the events that satisfy the queries are then retrieved. From the current time point in the playback, we locate the nearest synchronisation point in the log file of *frame buffer update* messages in the specified direction, i.e. either in the forward or backward direction, which corresponds to the timestamp of the closest instance of all the retrieved events. The screen image at that location is then drawn, and we can carry on searching for the next or previous event or start playing from there.

We have considered all the user and system events as potential indices, but our practice shows that not all the indices are equally useful. While *frame buffer update*, *key event*, *client cut text* and *server cut text* are among the most popular search items, *pointer events* are hardly referred to, because users don't usually remember where they clicked the mouse.

4. RELATED WORK

Many multimedia tools use digital audio, video and computing records to capture user activities [2], and automatically analyse the captured media using artificial intelligence techniques to derive indices or to create metadata that describes the media contents for subsequent access [3]. These techniques include speech processing, image processing and information retrieval.

While the traditional multimedia technologies support the capturing and indexing of user activities in general, our VNC technology focuses on computer-based activities only. We introduce a medium that captures computer-based activities with no need for extra devices such as cameras or video capture cards and extract user and system event information during the activities to produce a set of new indexing methods. As an example, the Informedia project generates key frames of a video segment by shot detection for significant colour shifts [6] and the MoCA (Movie Content Analysis) system generates video abstracts by calculating edge changes to detect scene breaks [1]. In the VNC system, the detection of screen changes is much easier: the *frame buffer update* implies the size of the frame buffer area that is changed in the update, so we can detect screen changes to generate thumbnail summaries of the recording without processing the screen images. As a second example, the DART (Digital Asset Retrieval Technology) project uses Optical Character Recognition (OCR) to identify probably text in video images [3]. The OCR mechanism can also be applied to screen images, and the result will be more accurate because screen text is easier to recognise. In addition to this mechanism, we also encourage users to highlight the text string of interest to identify screen characters. The disadvantage of this approach is that it requires the user to highlight the string in the heat of the moment and sometimes it is difficult for the user to decide whether a particular string can be a potential index. Our second approach is to unobtrusively capture users' key presses, because these key presses will most likely cause new characters to appear on the screen. The advantage of these alternative approaches is that they are less time-consuming and error-prone than OCR algorithms and can create indices from characters that appear on the screen at real time.

5. CONCLUSION

In this paper, we identified the importance of capturing and indexing user activities for the purpose of documenting work experience without inhibiting users' natural work practices. We distinguished computer-based activities from human-based activities. Computer-based activities are those activities that take place on the computer, such as opening an application window or editing an email message. We presented an ultra-thin-client computing technology that supports capturing of these activities with new indexing methods that traditional multimedia technologies do not provide.

6. REFERENCES

[1] Lienhart, R., Pfeiffer, S. and Effelsberg, W., "Video Abstracting", *Communications of ACM*, Dec. 1997

[2] Minneman, S., et. al., "A Confederation of Tools for Capturing and Accessing Collaborative Activity", *ACM Multimedia'95*, Page 523-534, November 1995.

[3] Pye, D., Hollinghurst, N., Mills, T. and Wood, K., "Audio-visual Segmentation for Content-based Retrieval", *The International Conference on Spoken Language Processing (ICSLP'98)*, Sydney, Australia, December 1998

[4] Richardson, T., Stafford-Fraser, Q., Wood, K. and Hopper, A., "Virtual Network Computing", *IEEE Internet Computing*, Vol. 2 No. 1, Jan./Feb. 1998

[5] Spiteri, M. and Bates, J., "An architecture to support storage and retrieval of events", *Proceedings of MIDDLEWARE 1998*, Lancaster, UK, September, 1998

[6] Wactlar, H., Christel, M., Gong, Y. and Hauptmann, A., "Lessons Learned from Building a Terabyte Digital Video Library", *IEEE Computer*, Page 66-73, February, 1999