WebNC: efficient sharing of web applications

Laurent Denoue
denoue@fxpal.com

John Adcock
adcock@fxpal.com

Scott Carter
carter@fxpal.com

Gene Golovchinsky
gene@fxpal.com

FX Palo Alto Laboratory Inc.
3400 Hillview Ave #4
Palo Alto, CA 94304

ABSTRACT
WebNC is a system for efficiently sharing, retrieving and viewing web applications. Unlike existing screencasting and screensharing tools, WebNC is optimized to work with web pages where a lot of scrolling happens. WebNC uses a tile-based encoding to capture, transmit and deliver web applications, and relies only on dynamic HTML and JavaScript. The resulting webcasts require very little bandwidth and are viewable on any modern web browser including Firefox and Internet Explorer as well as browsers on the iPhone and Android platforms.

Categories and Subject Descriptors
H.5.4 [INFORMATION INTERFACES AND PRESENTATION]: Navigation

Keywords
Web application sharing, web navigation, real-time collaboration, co-browsing

1. INTRODUCTION
Today, many applications run within a web browser. Yet it is hard for users to easily screen-share their web browser windows, for example during web-conferences or just to document an interaction they have in their browser. So we built WebNC ("web and see"), a screen-sharing system that focuses on sharing web browser windows, not the full desktop of the users.

We developed WebNC with an understanding of the nature of web applications and web pages and how they differ from typical desktop applications. Web interaction typically involves a lot of scrolling, mostly vertical but also horizontal, and contains mostly static elements like text paragraphs, but also multimedia elements such as images, video clips and animations. In addition, users typically go back and forth between pages: leveraging this behavior is important but unfortunately ignored by existing encoding techniques where video codecs use inter-frame similarity within a few milliseconds, leading to inefficient encoding of repeatedly viewed pages.

WebNC efficiently encodes web pages using less bandwidth than typical screen-recorders, including embedded multimedia elements such as applets, flash animations, movies, context menus, and tooltips. Pages are captured in full resolution but can still be delivered at lower resolutions. Users only require a modern web browser with JavaScript to view webcasts in real-time or retrieve them asynchronously: no additional plugin or ActiveX controls are required to view WebNC screencasts, only HTML and Javascript.

Finally, with WebNC, users do not need to reconfigure their networks or firewalls: WebNC uses only outbound HTTP 80 requests to the server, and the viewers only open outbound connections to the server for viewing webcasts.

Conference'09, Month 1–2, 2004, City, State, Country.

2. SYSTEM DESCRIPTION

2.1 Architecture
The system consists of 3 main components: 1) a publisher written as a browser extension for Firefox, 2) a central server that receives published content and serves it to 3) subscribers who can view webcasts by pointing their web browser to the server url with a unique sessionID they received from a publisher.
In order to work across all network configurations, publishers use standard outbound HTTP connections over port 80, and similarly, subscribers access the WebNC server over HTTP. This web page uses only HTML and JavaScript and is viewable on all modern web browsers, including on smartphones such as the iPhone and Android.

2.2 Sharing a web page
To start sharing a web page or web application, a user clicks on the WebNC icon on the bottom/right corner of the browser window after having installed the WebNC extension. The icon changes color to indicate that it is actively recording. The user is shown a sessionID that can be distributed to subscribers who wish to view the session live, or kept for later retrieval.

The extension starts capturing and sending the web page to the server, not requiring further user interaction. By default, WebNC only shares the browser tab that was active when the user clicked on the icon. If the user switches to another TAB in the browser, this new tab will not be shown

Living inside the web browser, the WebNC extension has access to the Document Object Model (DOM). At any time, it knows the vertical position of the scrollbar (it can also determine which elements have scrollbars, such as IFRAMEs, DIVs with scrolls, etc.) Using this value, it grabs a set of tiles (currently 256x256 pixels) starting from the top/left corner of the viewable area on the web page. It assigns each tile a number, starting from 0,0 for the top/left tile and moving right and down until it reaches the entire scrollable area (not only the viewable area).

Every 300ms, WebNC grabs the entire viewable area as a reference bitmap. It then compresses into PNG or JPG each tile until all tiles in the viewable area are done. The smaller image is hashed into a unique signature computed by appending the length of the encoded bits, an MD5 hash of the bits themselves, and the tile number. If the hash value has been seen before during that session, the tile content is not sent again to the server. This occurs commonly on the web when users navigate back to a previous page; all tiles will have already have been sent.

If a tile with the computed hash value has not yet been sent to the server, the actual image is sent along with its tile name (e.g. tile2,5), along with the mouse coordinate (curx, cury), the cursor shape (e.g. arrow, text beam, etc.), the size of the viewable area (vw,vh), the size of the entire scrollable area (sw,sh), and the current location of the scrollbar (scrollTop, scrollLeft).

WebNC also computes the bounding boxes of all plug-ins embedded in the page and its sub-frames, such as Acrobat Reader, Flash, QuickTime and Java applets. Each object is captured as its own tile (called a sprite). The corresponding area is cleared in the reference bitmap so that the tiling comparison does not detect changed pixels.

2.3 Viewing a web page

To view a WebNC session in real-time, a user simply opens the WebNC server's web page and provides the session identifier. Unlike the person who shared the web page, this user only needs a web browser with HTML and Javascript support: no special software installation is required. Android and iPhone smartphones are able to view the WebNC webcasts.

The HTML page uses Ajax requests to start receiving values for that session, including (vw,vh) to create a DIV element for the viewable area. Inside this DIV, the page places another DIV with a width and height set to the scrollable portion (sw,sh) sent by the publisher. The JavaScript code parses the list of tiles and positions them using absolute positioning inside this embedded DIV. Each tile is an IMG with its source pointing to the hash value. Finally, the position of the embedded DIV is set to the scrollLeft/scrollTop, giving users the effect of scrolling the viewport to the corresponding position. The mouse position (curx,cury) is used to place another IMG absolutely over the first DIV element. The IMG is chosen to replicate the original shape of the cursor.

Because the viewer code receives the data periodically from the server, WebNC uses timers to transition the locations of the reconstructed cursor as well as the scroll positions. This technique makes for a very smooth and enjoyable viewing experience.

3. EVALUATION

We built a prototype publisher as a Firefox extension in JavaScript that also uses a native C++ plug-in for Windows to capture objects such as Flash/QuickTime/Applets for which drawing surfaces were not available directly to the JavaScript extension. We built a WebNC server in Java that currently stores all tiles and accompanying data in memory, keyed under a given sessionid. We also produced a WebNC viewer in HTML/JavaScript that works on most desktop web browsers as well as mobile web browsers based on WebKit (iPhone/Android smartphones) and Internet Explorer for Windows Mobile.

Using WinMacro, we recorded a 2 minutes session of a 1024x768 pixels web browser window involving a few Google searches and reading of Wikipedia articles, including a fair amount of scrolling. WebNC used 252 kbps, versus 74 kbps for Microsoft Remote Desktop Protocol (RDP), 485 kbps for UltraVNC/TightVNC and 869 kbps for Microsoft SharedView. These results are encouraging: although RDP uses less bandwidth, WebNC does not require network reconfigurations, and most importantly does not require viewers to install additional plugins or ActiveX controls to view WebNC webcasts.

In addition to comparing with real-time screen-sharing tools, we also compared WebNC to Camtasia Studio, a well-known screen-recorder for Windows that uses a proprietary, specifically designed codec for screen capture. The same session lead to 1632 kbit/sec while encoding at 15 frames per second at the same resolution of 1024x768 pixels.

We also asked 3 users to use WebNC. They successfully shared their web window with a colleague or spouse. The experience was also seamless for viewers who were impressed by the smooth scrolling and mouse pointer motion. Two users suggested adding an instant messaging component and the ability to send their cursor location back to the person sharing their window. All three publishers wanted better visual feedback to show them exactly what they were sharing; a small preview window showing a thumbnail would probably suffice.

4. CONCLUSION AND FUTURE WORK

WebNC is a new tool for efficiently sharing web applications. Unlike existing systems, it leverages the page Document Object Model (DOM) to efficiently encode them and outputs the resulting recordings using standard HTML/JavaScript, making them viewable on most modern web browsers. By living inside web browsers as a plugin, it allows users to share only one tab and thus avoid privacy concerns typical of other tools when popup windows cover the shared viewport. We also showed how WebNC leverages the DOM to reproduce smooth mouse pointer motion and scrolling behavior that is most common while interacting on the web. Our first evaluation showed that WebNC faired well with respect to bandwidth requirements, and represent an important improvement by being accessible from different web browsers.

In future work, WebNC will use the accessibility interfaces available in all modern web browsers to extract words and their locations on the page. This feature will allow users to retrieve specific parts of webcasts containing a given keyword, and the player will be able to let users skim webcasts by showing them only the parts containing specific keywords. This component will also be able to filter out sensitive information like email addresses, social security numbers, phone numbers, and street addresses. Also, it would be interesting to support new query types based for example on whether webcasts contain video clips, images, form-filling activities, etc.

5. ACKNOWLEDGMENTS

Our thanks to FXPAL researchers who helped us evaluate WebNC and Larry Rowe and Lynn Wilcox for supporting this research