# Chronicle: Capture, Exploration, and Playback of Document Workflow Histories

*Tovi Grossman, Justin Matejka, George Fitzmaurice*

Autodesk Research
210 King St. East, Toronto, Ontario, Canada, M5A 1J7

**ABSTRACT**
We describe Chronicle, a new system that allows users to explore document workflow histories. Chronicle captures the entire video history of a graphical document, and provides links between the content and the relevant areas of the history. Users can indicate specific content of interest, and see the workflows, tools, and settings needed to reproduce the associated results, or to better understand how it was constructed to allow for informed modification. Thus, by storing the rich information regarding the document's history workflow, Chronicle makes any working document a potentially powerful learning tool. We outline some of the challenges surrounding the development of such a system, and then describe our implementation within an image editing application. A qualitative user study produced extremely encouraging results, as users unanimously found the system both useful and easy to use.

**Keywords:** Chronicle, History, Video, Workflow, Timeline.

**ACM Classification:** H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

**General terms:** Documentation, Design, Human Factors.

## INTRODUCTION

The majority of today's graphical user interfaces keep track of the operations that are carried out, to provide the ability to later undo their effects if necessary. Typically, this information is discarded when the document is saved. Unfortunately, lost with this history, is an abundance of what could be valuable information: the rich set of workflows which the user, or users, carried out to generate the document. Storing a document's workflow history, and providing tools for its visualization and exploration, could make any document a powerful learning tool.

For example, imagine opening an image file that a collaborator has sent you. You would like to apply an effect already existing in the image to another area. But, you are less skilled than your colleague, and are unsure how the existing effect was created. If the workflow history of that document had been stored and could be reviewed and accessed, you could learn exactly how that effect was applied, and be able to apply it yourself. *Despite their potential power, tools to help users learn from workflow histories have not been widely explored.*

In this paper, we describe Chronicle, a new system we have developed to support graphical document workflow exploration and playback (Figure 1). Chronicle allows users to browse graphical representations of a document's revisions, called *chronicles* (Figure 1a), and provides a visual scheme of state and event histories on an interactive timeline (Figure 1b). Once an area of interest has been located, the actual workflow can be played back in a full-resolution video (Figure 1c). Some of these components are inspired by previous history management and visualization tools (e.g. [13, 24, 25, 33]), but Chronicle integrates these features into a fully functional system within a real image-editing application. In addition, our technical contributions include the following novel features:

- Video capture of an entire document history, indexed by document revisions and UI events.
- Hierarchal clustering of a large-scale revision history.
- A rich set of tools to probe and filter the revisions.
- Visualizing high-level UI events within multiple streams of a workflow timeline.

After describing Chronicle, we report on a qualitative user evaluation, which produced extremely encouraging results, as users unanimously found the system both useful and easy to use.

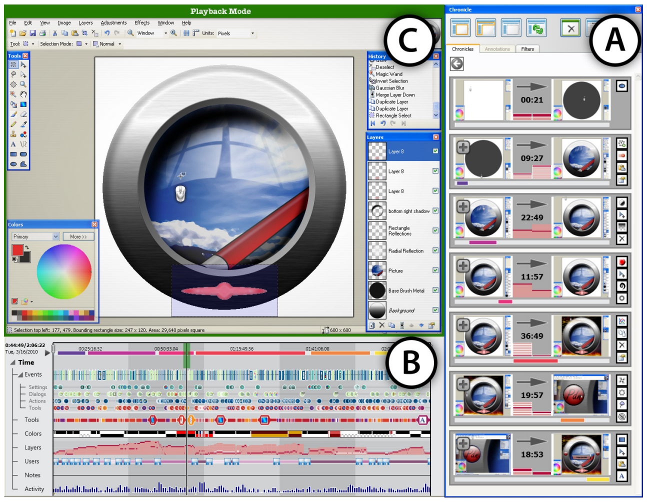


Figure 1. Chronicle. a) main Chronicle window, b) the timeline, c) application/Playback window.

## RELATED WORK

### Operation History and Undo Management

Providing efficient access to the history of operations a user has performed has been long explored in the HCI research. One of the most important aspects of these systems is how the operation history is visualized. This can include textual lists [6, 30], instructions presented in natural language [5, 26], enhanced screen shots [5, 33, 39], or before-and-after thumbnails [24, 25]. Recently, Nakamura et. al [33] presented the results of a study which showed that annotated graphical representations are an effective method for visualizing operation histories. Chronicle also uses graphical depictions of the revision history.

As documents become longer, with hundreds or thousands of commands, it becomes increasingly important to provide efficient access to the operations in the history. Crystal [31] allows users to ask questions about specific content in a document, but its focus is more on explaining why something did or did not happen, rather than describing previous operations or workflows. Hardock et al. supports "undo by selection" where simple operations can be accessed by directly selecting its associated object in the document [16]. Both Meng et al. [28] and Nakamura et al. [33] allow users to filter the operation history to only show events that affected specific areas of the documents. However, only basic documents were considered. For example, the study presented in Nakamura et al. tested a document lasting 150 seconds with 50 operations [33].

The Chimera system [22, 24] clusters operations into groups at a user-defined level of granularity, but the coarsest level only collapses similar operations together. Nakamura et al. also provides low-level clustering [33]. Without higher-level clustering, long-term documents would have long operation histories to search. Chronicle extends these systems by providing probing and filtering mechanisms, as well as a hierarchal clustering algorithm.

Operations histories have been leveraged for a wide spectrum of purposes, including revision branching [7, 10], macros definition [22, 25], and tutorial generation [5]. Some research [23] and commercial systems (such as Dassault Systèmes SolidWorks and Autodesk Inventor), allow modification of past events propagate up to the current version of the document. Heer et al. [18] use graphical histories to support analysis in information visualization systems, while Experiscope [15] visualizes low-level operation histories on a timeline to support analysis of empirical studies. Our work builds upon these previous systems, however our focus is to support users to understand and learn higher-level workflows. This is in-part inspired by the previous work by Plaisant et al. who outline the numerous potential benefits of "learning histories", including "facilitate collaborative learning" [35].

### Time Travel and Visualization

A number of applications have been developed to allow users to "travel" back in time. Flatland provides "semantic time snapping" allowing users to attach a time slider to any object within a whiteboard application [32]. The Visual Knowledge Builder provides a similar timeline interface for viewing the authoring history of hypertext documents [38]. Timescape also uses a timeline metaphor to organize objects on a computer desktop, providing the user with a slider to restore earlier states of the desktop's appearance [37]. Lifelines provides rich interactive timeline views for personal life events, such as medical histories [34]. Chronicle builds upon these systems but for the viewing and playback of interaction workflows within a GUI.

### Video Summarization and Browsing

Numerous projects have aimed at the general challenge of summarizing and browsing large-scale videos. Many techniques have been developed, including elastic speed playback [36], browsing by direct manipulation [12], still-image abstraction [27], animated video skims [9], and adaptive fast-forwarding [8]. A full review of such systems is beyond the scope of our work, and we refer the reader to Truong et al. who summarize and classify the previous work in this area [41]. Almost all of related work focuses on the summarization of live motion videos, and not UI screen captures, which are much more structured. Chronicle explores how such information can be leveraged.

### Multimedia Tutorials

In recent years, new forms of multimedia learning aids have arisen. Stencil-based tutorials [20], and Graphstracts [19], both use graphical visual overlays to help users understand steps in tutorials. Sikuli [42] proposes a vision-based method to automate creation of such materials. Photo-Manipulation Tutorials [13] are made automatically from user operation histories. Animated demonstrations have also been used [17], and a recent study by Grossman et al. showed that short video clips can be effective learning aids [14]. However, for longer videos, aids for navigation will be essential [17], and Chronicle thus supports a suite of techniques for navigating the space of revisions.

A few commercial products, such as Microsoft's now expired CommunityClips[1] project, and Corel[2] Painter's session recording, allow users to capture and create application usage videos. These tools make the recording of video screen captures more convenient, but no aids for exploring the captured workflows are provided, other than navigating through the produced videos.

In summary, there have been numerous previous projects related to Chronicle. However none that we are aware of explicitly studied the use of operational histories to help users understand the workflows used, and we are also unaware of any previous systems which log and summarize the *video history* of a document and link that video to the content of the document and UI components of the interface. In the following, we will be presenting a new system, Chronicle, which does so, and describe how we designed the tool to address the unique challenges which arise from these new properties.

[1] www.communityclips.officelabs.com [2] www.corel.com

## CHRONICLE

The basic idea of Chronicle is to give users the ability to see the actual video of how any of the content within the document was created or modified. This idea builds upon previous work in capture and indexing of multimedia document content [1, 21, 29]. The prototype was implemented by instrumenting Paint.NET, an open source raster-based image editing application. This application was chosen as a suitable GUI application that could be modified. However, our goal is for the Chronicle concepts to generalize to other application domains as well.

### System Overview

The system can be divided into three main components. The main Chronicle window (Figure 1a) is displayed to the right of the application window, an interactive timeline is displayed below the application window (Figure 1b), and, the application window itself is used to display the current document and also playback captured video (Figure 1c).

### Chronicles

Like previous systems [24, 25], the main Chronicle window shows a list of before (Figure 2a) and after (Figure 2b) thumbnails, called *chronicles*, each representing a revision. However, *chronicles* also contain workflow information. A set of tool icons represents the tools used within that time segment (Figure 2c). A maximum of four tools are shown, chosen as the most unique tools across the entire history. Before and after layer information is displayed (Figure 2d), since sometimes the thumbnails do not change, but the layer information does. The layer information indicates the number of layers, which layer was active (dark red), and which layers were invisible (white). A tooltip for the layer information shows what the actual layers palette looked like at the associated time.

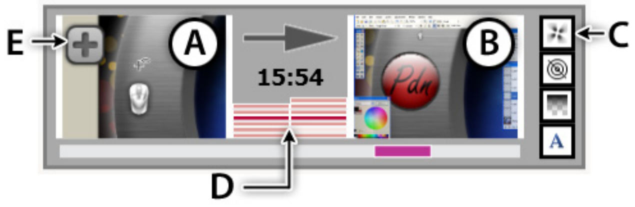


Figure 2. A *chronicle*. a) before and b) after thumbnails. c) Icons represent the tools used in the revisions. d) layer information. e) hierarchy button.

### Chronicle Hierarchy

We extend previous operation grouping [22] into a new hierarchal clustering system, which enables support for large-scale histories. The *chronicles* are organized into a hierarchy, so that no more than 7 are displayed at a time, which prevents the need for scrolling. The clustering algorithm will be discussed in the System Implementation section. Any *chronicle* that contains multiple operations can be expanded, and this hierarchy is represented with an icon on the before thumbnail (Figure 2e). Clicking on it generates a new list of at most 7 revisions, each of which occurred within the time represented by the parent *chronicle*. This process can be repeated until each *chronicle* has a single operation. The user can return to a higher level in the hierarchy by clicking a back button. This hierarchal navigation provides users with an initial higher-level view of revisions, but also efficient access to detailed operations.

### Video Playback

A main contribution of our system is the ability to view the actual video from when content was authored. Clicking on a *chronicle* will start the video playback of the associated revision, beginning 5 seconds before the revision's actual start time. The video is displayed as an overlay on top of the main application at full resolution, to reveal the interaction and interface details. Overlaying the video on top of the UI provides a seamless transition to the playback mode. An overlay skin, with a green border and faint horizontal scan lines (Figure 3), is displayed to ensure the users understand that they are seeing video content and not a live document. Hitting escape, or clicking a "clear video" button (Figure 4e), will clear the video overlay.

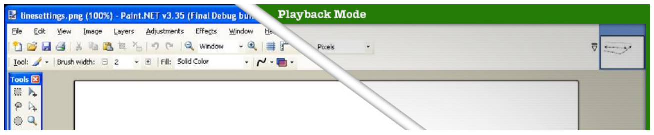


Figure 3. In playback mode, the video is overlaid directly on top of the main application, with a green skin indicating the mode.

### Probing Tools

Another main novel feature of Chronicle is a suite of probes to aid navigation of the revisions. Three probing tools are available to indicate content or UI controls of interest (Figure 4a-c). The probes build upon previous content-based history mechanisms [16, 28, 32, 33, 38].

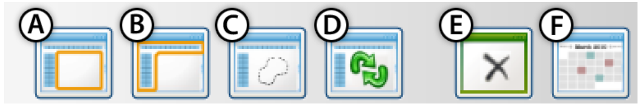


Figure 4. Chronicle UI controls: a) Data probe. b) UI probe. c) Selection probe. d) Refresh Revisions. e) Clear Video. f) Calendar View.

#### *Data Probe*

The Data Probe is used to specify regions of interest within the document content. It is displayed as a grey semi-transparent square (Figure 5a). Its size can be adjusted with the scrollwheel and its position is controlled by the mouse. Clicking the mouse button generates *chronicles* associated with the underlying document content. When in playback mode, the data probe can be used on the video.

By holding down the control button, the data probe becomes a temporal lens [2], allowing users to get a quick glimpse into the past in only a specific region (Figure 5b). Using the scroll wheel, the user can scroll forwards and backwards in time. This "preview" time is indicated on the timeline with a red vertical line, and the preview time automatically skips to the points in time when revisions occurred in that region. Moving the lens reveals what other areas of the document looked like at that time.

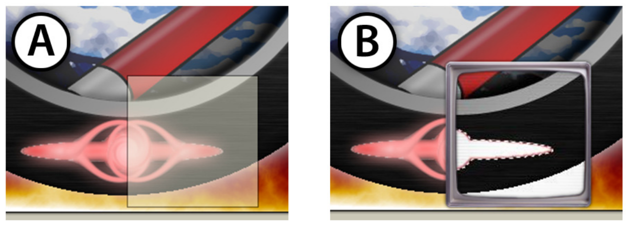


Figure 5. a) Data probe. b) Holding down the control key changes the probe into an in-place lens revealing the previous states under its region, and the user can scroll forwards and backwards in time.

### UI Probe

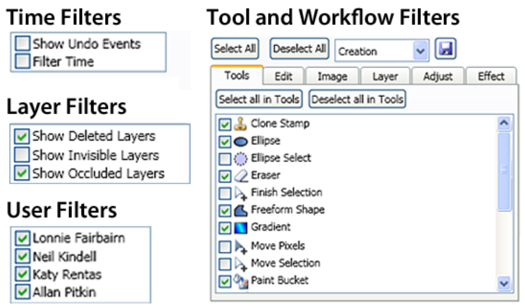
Unlike previous system, Chronicle also allows users to probe application UI controls, through the UI probe. This probe works the same as the data probe, except it is yellow, and considers only the UI components within its region. The UI probe can be used on any of the individual tool icons, floating palettes, setting icons, and menus. Clicking on any of these items with the UI probe refreshes the *chronicles* to show when those specified UI components were used. When probing a menu, the revisions include any submenu items that are a child of the menu. To probe a specific menu item, the user first highlights the item, and then hits F2 (which is the shortcut key for the UI probe).

### Selection Probe

The third probe uses the application's own selection tools, allowing for a more detailed specification of the region of interest. The "*probe by selection*" icon generates *chronicles* based on the region currently selected. If there is no current selection, then the entire workspace is probed.

### Filters

While filters are typical components for exploring information [18], we provide a set of novel filters to specifically aid the access of large-scale document workflow histories. These filter controls are in a tab of the main Chronicle Window (Figure 6).



Time Filters
- ☐ Show Undo Events
- ☐ Filter Time

Layer Filters
- ✓ Show Deleted Layers
- ☐ Show Invisible Layers
- ✓ Show Occluded Layers

User Filters
- ✓ Lonnie Fairbairn
- ✓ Neil Kindell
- ✓ Katy Rentas
- ✓ Allan Pitkin

Tool and Workflow Filters
Creation
- ✓ Clone Stamp
- ✓ Ellipse
- ☐ Ellipse Select
- ✓ Eraser
- ☐ Finish Selection
- ✓ Freeform Shape
- ✓ Gradient
- ☐ Move Pixels
- ☐ Move Selection
- ✓ Paint Bucket

Figure 6. The filter tab provides UI controls to filter by time, layers, users, tools, and workflows.

#### *Time Filters*

In a larger document, there becomes a risk of including extraneous content in the *chronicles* and video content. While idle times will be automatically dismissed by our clustering algorithm, we do provide controls to allow users to specify particular times of interest by enabling "filter by time". The user specifies a time of interest using the timeline's zoom sliders. The *chronicles* will then be updated to only show results within that interval. The undo filter sets Chronicle to ignore any action that has been reversed by an undo action. The time and undo filters only affect the generation of *chronicles* - the associated video content is not removed, and can still be accessed by scrubbing through the video.

#### *Layer Filters*

We designed Chronicle to be layer-aware, given the importance that layers have within image editing applications. Filtering deleted layers ignores revisions that occur on deleted layers. Filtering occluded layers ignores revisions within regions that are fully occluded by other layers. Filtering invisible layers ignores revisions that are not currently visible, either because the associated region is fully transparent, or its layer is currently set to be hidden by the user. Thus, users can use the application's existing layer palette to specify a layer of interest, and then generate *chronicles* for that individual layer.

#### *User Filters*

If a document has more than one author, the user filters can be used to show the revisions made by a specific subset of those authors.

#### *Filtering by Tools and Workflows*

A tool and workflow filters manager allows users to select any individual feature within the Paint.NET system to filter. The features are organized into categories, such as tools, layers, and effects. The user can select or deselect any individual item, an entire category, or the entire set across all categories. This gives users full control over the *chronicles* that will be generated. For example, a user may want to see all the effects that were applied in a specific area of the document. Inspired by Terry et al.'s "task sets" for ingimp [40], a drop down box can be used to select from preset "workflows" which have an associated set of features. For example, the "Creations" workflow selects only the tools and effects that can be used to add content to the document. Chronicle also has a preset workflow called "Tools I've Never Used". We mocked up this category by associating some of the more obscure commands with it, but in an actual deployment, Chronicle could keep a record of the tools an end user has not used. This could help users identify learning opportunities. Users can also save their own tool sets by clicking a save icon in the manager.

#### *Generating Filtered Chronicles*

All of the described filters can be used in combination with the probing tools. Once the filters are set, the user can probe a new region or refresh the changes within an already specified region (Figure 4d). The user can restore the default filter settings with a button on the filter tab.

### Annotations

In some scenarios, an author may intentionally generate a Chronicle document with the knowledge that it will be used by other users as a learning resource. In such scenarios, that content author may wish to add annotations to the

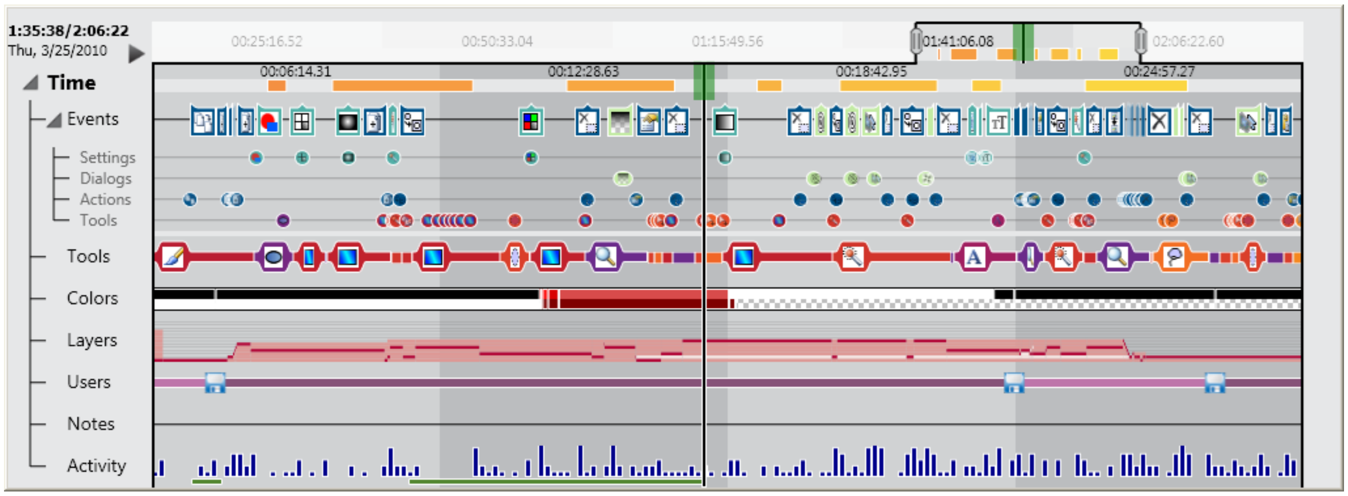


Figure 7. The expanded timeline, showing a zoomed in view after expanding one of the *chronicles*. This view represents 25 minutes of usage time.

document. Authors can do this by going to a specific frame in the video, positioning the cursor over the desired region, and then hitting a hotkey (F8) to add an annotation. The authors can then type the annotation. This is similar to video annotations [36], but the annotations are content-aware, so that they can be displayed in an annotation tab of the Chronicle window when the associated region is probed.

## The Timeline

Like previous systems [33, 34, 38], we provide a timeline to visualize and control playback of past workflows. However, we introduce a rich visual language which has been specifically designed to illustrate high-level and low-level workflow information. A green playback handle can be dragged to scrub the video in real time. In the top row of the timeline, colored strips indicate the location of the current *chronicles*. To support large-scale document histories, two zoom handles can be used to indicate an area of interest. When these handles are set, a second, local timeline, is displayed below the top, global, timeline. When the user expands a *chronicle*, the zoom handles are automatically repositioned to focus on the region associated with that *chronicle* (Figure 7).

Based on user evaluations with early prototypes, we realized it was important to include an indication of key events within the timeline and a method to navigate to those events. Without such information, users may not be able to find all the information required to understand how a task was completed. For example, a setting change or selection may have occurred prior to that actual revision occurring. An expanded view of the timeline, separates the timeline into a number of "tracks" each showing its own stream of information. Each track contains markers that can be clicked to seek to the associated time in the video. Each track also has its own tooltips that provide additional information when the user hovers over a marker (Figure 8).

### *Events Track*

The events track is similar to an entire history list of the document, mapped onto the timeline. The one difference is that the event list also stores setting changes and tool configurations, which typically are not stored in a document's history. Each event is represented by a marker that has a variable width to prevent overlap. The events track can be further expanded into four detailed tracks, which categorizes each event. That is, for each event, there will be a marker on the main event track, and a marker on one of the four detailed tracks. An exception is made for "tool" events, which are not shown on the main event track, because that information is represented on a separate main tools track below. The markers on the detailed tracks are displayed as small dots with their associated icons. If tool filters have been applied, then markers will only appear if they are associated with the current filter criteria.

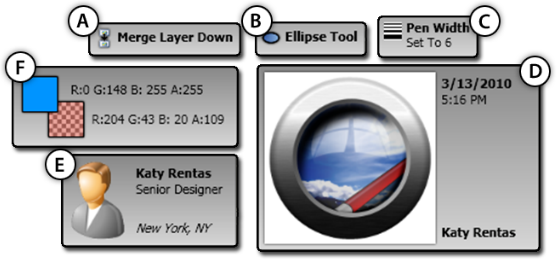


Figure 8. Sample tooltips for: a) action marker, b) tool marker, c) setting marker, d) save point, e) user marker, f) color marker.

*Settings sub-track:* Light blue setting markers represent whenever a setting changed. Their tooltip will display the exact setting and value (Figure 8c).

*Dialogs sub-track:* Dialog markers represent whenever a modal pop-up dialog box was used. The markers are actually rounded rectangles, representing the time when a dialog was open. The markers are color-coded light green. Their tooltips will show the actual dialog box using a phosphor effect [4] (Figure 9). The phosphor effect is added completely automatically without any knowledge of the contents of the dialog. This is accomplished by taking a difference between the start and end image of the dialog, and highlighting any changes on top of the final image with a red glow. This allows us to provide phosphor effects for non-typical UI components such as the circle seen in Figure 9. When the tooltip is open, the user can hit the spacebar to cycle through the before and after images.

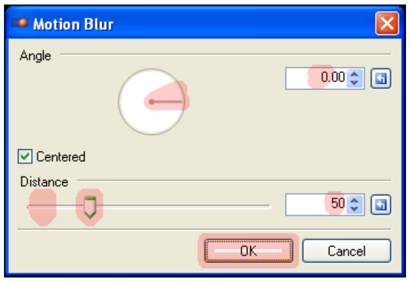


Figure 9. Tooltip for a dialog marker. A red phosphor effect is used to highlight which of the values within the dialog box have been changed.

*Actions sub-track:* The actions sub-track contains events executed from clicking on menus items or icons within the user interface, such as adding or deleting a layer. They are color-coded dark blue.

*Tools sub-track:* Tool markers show when the current tool (such as the paintbrush) is used. Each tool is individually color-coded. When the cursor hovers over a tool marker, markers for any prior settings that affected the behavior of that tool are highlighted. If an associated setting is out of view, a halo [3] is displayed to indicate its existence (Figure 10). The user can right click a tool marker, and choose "highlight all relevant settings" from a contextual menu, to adjust the timeline zoom handles so that each relevant setting is in view. The setting-tool associations were hard-coded into the system, but previous work has proposed application frameworks to support such associations with little overhead costs [31].

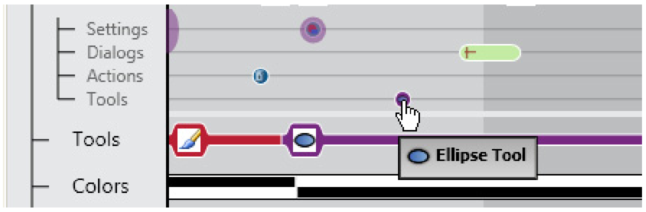


Figure 10. When the cursor hovers over the ellipse tool event marker, previous setting events which effected that tool are highlighted. A halo [3] indicates the existence of an additional relevant setting marker currently out of view.

### *Tools Track*

The main tools track shows what the main tool was set to at any point in time, using the same color scheme of the tool events directly above it. This track represents an application state, rather than an explicit event.

### *Colors Track*

The colors track shows what the primary (top) and secondary (bottom) colors were set to. Alphas values are indicated by placing a checkered pattern behind the color markers. The color marker tooltips specify the exact ARGB values of the colors (Figure 8f).

### *Layers Track*

The layers track provides the current layer information for the document, using the same visual encoding as in the *chronicles*. For example, in Figure 7, we can see that the layers were built up over time, until later when they were flattened. The tooltip for the layers shows what the actual layers palette looked like at the associated point in time.

### *Users Track*

The users track indicates the current user at any point in time. Each user is assigned a unique color. The tooltip for a user marker shows information about that user (Figure 8e). This track also contains save icons, indicating when the document was saved. The tooltip for the save icons shows what the document looked like when saved (Figure 8d).

### *Notes Track*

The notes track shows the annotations that have been added to the document. The annotation markers are color coded based on the user who created them. The markers can be dragged in the timeline to adjust their position or duration.

### *Activity Track*

The final track is used to give a sense of how active the user was at any point in time. The timeline is divided into vertical bars with a height to indicate how many mouse events occurred at that interval of time. Horizontal green lines indicate portions of the history that have been viewed.

## Marker Interactivity

A potential benefit of Chronicle is that it would allow a user to view previous states of a document that they were currently working with, and potentially remind themselves of how they accessed certain features, or how certain tools were previously configured. We embraced this usage scenario by providing a link between the timeline markers and the main application. For example, right clicking a dialog box marker pops-up the actual dialog, which can be used for their current work. In the tool marker's contextual menu, the user can select "match all relevant settings", which automatically configures the user's actual tool. Alternatively, an individual setting can be matched by right clicking its marker. Right clicking on a color marker shows a context menu where the user can choose to set their own palette colors to the colors used at that point in time. These tools are similar to a "format painter", but used across time.

## Fast Forward Playback

It is unlikely a user would want to watch the entire video, but users may want to quickly review specific areas that are of interest. To do this, the user can first set up the desired filters, and then click directly on the label of the events track or its sub-tracks. This will begin dynamic speed playback where the markers on the associated tracks act as "semantic event points" [8]. The video will play 2 seconds before and after semantic event points at 1.5x speed, and will quickly fast forward to reach the next event point.

### Calendar View

Another way to see which users have been working with the document, and when the document was saved, is a "calendar view", which shows what the image looked like at the end of each day it was worked on. The same tooltips as the save markers are used for these images (Figure 8d). Hovering over any day will highlight the relevant area in the timeline, and clicking on an image will seek the video to the beginning of that day.

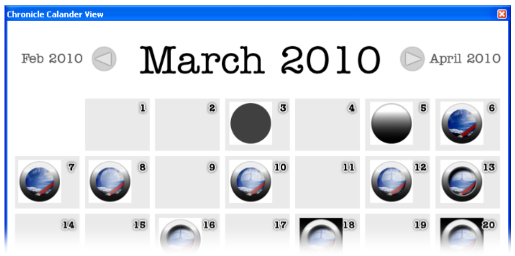


Figure 11. The calendar view shows what the document looked like at the end of each day. Note: The illustrated document was created across a span of 4 days, and we manually altered each save date so that we could prototype this view.

## SYSTEM IMPLEMENTATION

### System Software and Hardware

We instrumented Paint.NET to internally log all document edits and UI events. This was accomplished with minimal complexity by modifying the existing history stack management system. Video content is displayed using the Windows Presentation Foundation and captured using the "VH Screen Capture Driver" DirectShow filter and the DirectShow.NET library. All of these components were integrated into the original Paint.NET Visual Studio C# Solution. Both the development and implementation occurred on a dual core 2.4GHz Windows XP machine with 2 GB of RAM.

### System Architecture of Capture Facilities

We now describe the system architecture used to implement the required capture facilities for Chronicle. The most significant properties of our capture architecture is that it runs automatically, in real time, and is invisible to the content author.

#### *Capturing Revisions, Events and Images*

We store an array of changes for *cells*, where a cell is an n x n grid of pixels. Optimally, n would equal 1, but for high resolution documents the memory loading and computation time becomes too costly. For an 800x600 image, a 4x4 cell size worked well. After every user operation, we query, in a separate thread, every pixel (regardless of cell size) on the active layer, to see if its ARGB value has changed. If it has, we add the current *document time* and the active layer *ID* as a (*time*, *ID*) pair to the cell's array of changes. The document time is the total running time it has been open across its entire history.

We timestamp and store all the events that occur in an array. In addition to the operations that would typically appear in a history list, this array stores times for tool changes, color changes, setting changes, save events, mouse clicks, and document navigations (zooming and panning). Each event type has its own associated meta-data. For example, a color change event stores the primary and secondary color, and a save event stores the current user name and actual date and time of the save event.

In addition to storing these events, we also store the layer states of the document (which layer is active, how many layers there are, and which layers are invisible). This also includes storing the *IDs* of layers that have been merged together, so that appropriate *chronicles* are generated when a merged layer is probed.

Finally, we store static images: of dialogues when they are opened and closed; of the layer palette when it changes; and, of the entire document when it is saved.

#### *Capturing Videos*

The main Paint.NET application window size is fixed at 1024x768, and this entire screen region is captured at full resolution at 10fps. We use Techsmith's TSCC lossless codec to capture the video. Videos are captured in one minute segments so they can be processed by adding an overlay of a mouse icon indicating cursor position and button-presses, and any keyboard key-presses that occurred. The video processing happens on a separate thread, with each 1 minute segment taking approximately 15 seconds to process. Thus, regardless of the total capture time, the Chronicle window can be launched 15 seconds after the last video capture has ended.

#### *Generating the Chronicles*

The first step in generating the *chronicles* is to create a list of change times based on the current probe region and current filter parameters. For every cell in this region, the system iterates through the cell's array of changes. For each (*time, ID*) pair we test if it passes the current filter criteria. If it does, it is added to the list of change times. Once the list of change times is generated, it is sorted and passed to our hierarchal clustering algorithm.

#### *Hierarchal Clustering Algorithm*

The algorithm takes as input the list of change times, *InputTimes*, as described above, and then chooses at most six points in time to serve as delimiters between clusters.

Our experimentations indicated that using save times would be the best method to delimit the clusters, since in most cases users save their document after completing a coherent set of steps. As such, the algorithm first only considers the times $t$ in *InputList* that immediately precede a save event. If more than six such times are in the list, then the algorithm sorts the save events by the *absolute time* until the next event after the save occurred. For example, a save event for which the document was also closed and then not revised for another two days would be chosen before a save event where the user immediately began working again.

If less than six delimiters are generated from save events, then the algorithm goes on to consider every other time $t_i$ in

*InputList*, and sorts them by the time until the subsequent *InputList* time. That is, time $t_i$ would be sorted higher than time $t_j$ if and only if $t_{i+1} - t_i > t_{j+1} - t_j$. The algorithm picks from the top of this sorted list of times, until the maximum six times have been selected.

*Chronicles* are generated based on the delimiter times chosen. Similar to previous work [23, 33], if the entire workspace is probed, the thumbnail images are automatically cropped to only show the regions of the document that changed within the time interval represented by that *chronicle*. If the data probe was used, the thumbnail images are cropped images showing the probe region, which gives the user a more detailed zoomed in view.

#### *Saving and Loading Chronicle Documents*

When a user saves a Chronicle document, the data and images which have been captured are written to a directory associated with the file, and the video file is also moved to this directory. When a file is loaded into our application, we check for the existence of this associated directory. If it exists, the Chronicle document information is loaded.

## EVALUATION

We conducted a qualitative user study to evaluate the design aspects and features of Chronicle. We recruited eight external participants from an online posting, all with at least 3 years of experience with image editing software.

The evaluation sessions began with a 5 minute introduction to the system. This was followed by a full walkthrough of the system that lasted approximately 30 minutes. During this walkthrough, the users were first shown individual features, and asked to accomplish 26 "atomic" tasks. Example tasks included "probe the region where the white circles are" and "from viewing the save events on the timeline, which user created the white circles?" The walkthrough was conducted using a document with a 45 minute history. After the full walkthrough, the image illustrated throughout this paper, which had a total document time of just over 2 hours was loaded. The user was asked to independently complete 5 challenge tasks:

1. *How did the fire get its color?*
2. *What is the radius of the Gaussian Blur around the perimeter of the logo?*
3. *What font size was used for the left and right "Pdn"?*
4. *Find when the "add noise" effect was used.*
5. *Which blurring tool was used most often?*

Before completing these tasks the users were reminded of the main features of the system. The challenge tasks took approximately 10 minutes to complete. These challenge tasks were used to get a sense of how well Chronicle could be used with a real document, not how well new users could learn to use Chronicle.

After completing the challenge tasks, the participant filled out a 10-minute questionnaire. For each individual feature, the questionnaire asked them to rate, on a 7-point Likert scale, the statements "I found it easy to use" and "I think it would be useful (1 = poor, 7 = excellent).

### Results

The results of the study were very positive. Users were genuinely impressed with the demonstrated features, and thought that the tool would be quite useful. A few users (3/8) commented that it was a bit overwhelming to try to learn everything in such a short period, but they also thought that it would not take too long to master all of the features. In the full walkthrough, users completed 92% of the tasks independently after instruction.

For the challenge tasks, each of the users were able to complete all 5 tasks in under 2 minutes. Mean completion times for the 5 tasks were 35s, 54s, 34s, 22s, and 27s respectively. These times are encouraging, considering that the entire document history was over 2 hours long. Task 2 had a slightly higher average completion time because 2 of the users forgot to zoom in as far as possible to the relevant area in the timeline. As a result, the timeline markers were more dense then they had to be, and the users had trouble finding the right dialog box marker. The average time for the remaining 6 users for this task was 37s, and one user was able to complete this task in 12 seconds.

The questionnaire also elicited extremely positive results. Averaged across all features, average rankings were 6.2 for *easy to use* and 6.4 for *would be useful* (all scores out of 7). The final question was to rank the overall system. Average responses for the overall system were 6.1 for *easy to use* and 6.9 (all 7 except for one 6) for *would be useful.* Users consistently commented that they thought the tool would be extremely useful.

The features users found most useful were: using the data probe to specify areas of interest (6.8); the ability to filter by specific tools or preset workflows (6.6); the ability to see settings that were associated to the usage of a tool (6.8); and, the ability to match settings and colors to those used in the document (6.9).

Only 3 features got less than 6 for *easy to use*. The tools icons in the *chronicles* (5.8) were sometimes misleading because they were not necessarily exhaustive of all tools used in that associated time. The UI probe (5.5) was found to be awkward by one of the users, because the shape of the probe didn't always match the shape of the UI component he was trying to probe. And, the in-place playback within the data-probe was found to be tricky to use by two users.

Overwhelmingly positive comments were made by the participants throughout the study. One user, who used to do graphics design for a newspaper said "this is so cool - for my work as many as 5 or 6 people work on a document and this would really help", and "I can look at someone's professional work and steal their skills". A professional graphic designer, who had been using image editing tools for over 15 years, said "I've never encountered anything like this … this is incredible". Another graphics designer said "I'm really impressed that it not only shows the work history it also shows you a tutorial – I really like how the video is stored", and during the questionnaire said "I know it's going to sound strange but I like it all, I really do."

## DISCUSSION

Based on our own experiences, and comments made by participants of our evaluation, we believe Chronicle could be a valuable tool for a number of scenarios:

*Team Support*: A member of a team could see the revisions another member made to a shared document, to understand exactly how the tasks were carried out.

*Implicit Learning Aid*: A user could download a public document, and see exactly how the main workflow was performed and learn any new features that were used.

*New Tutorial Format*: As an easier alternative to authoring carefully crafted tutorials, an instructor or company could simply generate and post sample documents, which users could review to understand the workflows used to create the document.

*Self-Retrospect*: A user could see history of their document to remind themselves how they did something in the past, or re-instate a previous setting for increased productivity.

We have implemented Chronicle within an image editing application, but we do not see any intrinsic barriers preventing the adaptation of the higher-level concepts to more general software domains. Instead of storing changes for "cells," Chronicle would store changes for whatever "entities" make up the content of the target application. For example, in a vector-based application, the history of each individual geometry element would be stored, and could then be subsequently probed. Instead of querying the object's ARGB values, Chronicle would query the entities to see if any of their defining properties have changed.

## FUTURE WORK AND CONCLUSION

Our work opens up new opportunities for empirical research and intellectual advancement. We hope to perform future studies to investigate if Chronicle does indeed support knowledge transfer, and allow users to learn new skills and understand workflows from an existing document history. Our own experiences and initial pilot studies give promise to these hypotheses, but evaluating the system in more realistic usage scenarios could formally indentify how the system is used for learning purposes, productivity enhancements, and collaboration facilitation.

One obvious issue regarding a deployment of Chronicle is memory consumption. The video size of the 2 hour document we used is 1.3Gb (the remaining meta-data is only 12Mb compressed). Potential alternatives to storing local videos are streaming video content to and from a network server, or recording scripts instead of raw videos, which would significantly reduce storage size.

Finally, our implementation is application-dependant. Instrumenting the source code gave us access to pixel color values and history event streams. In contrast, an application independent operation browser was recently developed by Nakamura et al. [33]. Their implementation ideas, in combination with the work of Dixon and Fogarty's Prefab system [11] and Yeh et al.'s Sikuli system [42], which automatically recognize UI components, could be adapted to implement Chronicle in an application independent fashion. This could also allow existing video tutorials to be "scraped" and transformed into Chronicle documents.

To conclude, we have introduced Chronicle, a new type of system that supports the review of a graphical document's workflow to help understand how aspects of the document were constructed. By linking contents of the document, and components of the UI, directly to video playback, users can directly navigate to areas of interest and watch all the detailed subtleties of the interaction take place. Feedback from our evaluation was unanimously positive, strengthening our belief that Chronicle could change the way we think about what a document is.

## REFERENCES

1. Abowd, G. D., Atkeson, C. G., Feinstein, A., Hmelo, C., Kooper, R., Long, S., Sawhney, N. and Tani, M. (1996). Teaching and learning as multimedia authoring: the classroom 2000 project. *ACM Multimedia.* 187-198.
2. Adar, E., Dontcheva, M., Fogarty, J. and Weld, D. S. (2008). Zoetrope: interacting with the ephemeral web. *ACM UIST.* 239-248.
3. Baudisch, P. and Rosenholtz, R. (2003). Halo: a technique for visualizing off-screen objects. *ACM CHI.* 481-488.
4. Baudisch, P., Tan, D., Collomb, M., Robbins, D., Hinckley, K., Agrawala, M., Zhao, S. and Ramos, G. (2006). Phosphor: explaining transitions in the user interface using afterglow effects. *ACM UIST.* 169-178.
5. Bergman, L., Castelli, V., Lau, T. and Oblinger, D. (2005). DocWizards: a system for authoring follow-me documentation wizards. *ACM UIST.* 191-200.
6. Berlage, T. (1994). A selective undo mechanism for graphical user interfaces based on command objects. *ACM Transactions on Computer-Human Interaction.* 1(3):269-294.
7. Callahan, S. P., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T. and Vo, H. T. (2006). Managing the Evolution of Dataflows with VisTrails. *ACM SIGMOD.* 745-747.
8. Cheng, K.-Y., Luo, S.-J., Chen, B.-Y. and Chu, H.-H. (2009). SmartPlayer: user-centric video fast-forwarding. *ACM CHI.* 789-798.
9. Christel, M. G., Smith, M. A., Taylor, C. R. and Winkler, D. B. (1998). Evolving video skims into useful multimedia abstractions. *ACM CHI.* 171-178.
10. Derthick, M. and Roth, S. F. (2001). Enhancing data exploration with a branching history of user operations. *Knowledge-Based Systems.* 14(1-2):65-74.
11. Dixon, M. and Fogarty, J. (2010). Prefab: Implementing Advanced Behaviors Using Pixel-Based Reverse Engineering of Interface Structure. *ACM CHI.* 1525-1534.

12. Dragicevic, P., Ramos, G., Bibliowitcz, J., Nowrouzezahrai, D., Balakrishnan, R. and Singh, K. (2008). Video browsing by direct manipulation. *ACM CHI*. 237-246.

13. Grabler, F., Agrawala, M., Li, W., Dontcheva, M. and Igarashi, T. (2009). Generating photo manipulation tutorials by demonstration. *ACM SIGGRAPH*. 1-9.

14. Grossman, T. and Fitzmaurice, G. (2010). ToolClips: An Investigation of Contextual Video Assistance for Functionality Understanding. *ACM CHI*. 1515-1524.

15. Guimbretiere, F., Dixon, M. and Hinckley, K. (2007). ExperiScope: an analysis tool for interaction data. *ACM CHI*. 1333-1342.

16. Hardock, G., Kurtenbach, G. and Buxton, W. (1993). A marking based interface for collaborative writing. *ACM UIST*. 259-266.

17. Harrison, S. M. (1995). A comparison of still, animated, or nonillustrated on-line help with written or spoken instructions in a graphical user interface. *ACM CHI*. 82-89.

18. Heer, J., Mackinlay, J., Stolte, C. and Agrawala, M. (2008). Graphical Histories for Visualization: Supporting Analysis, Communication, and Evaluation. *IEEE Transactions on Visualization and Computer Graphics*. 14(6):1189-1196.

19. Huang, J. and Twidale, M. B. (2007). Graphstract: minimal graphical help for computers. *ACM UIST*. 203-212.

20. Kelleher, C. and Pausch, R. (2005). Stencils-based tutorials: design and evaluation. *ACM CHI*. 541-550.

21. Klemmer, S. R., Thomsen, M., Phelps-Goodman, E., Lee, R. and Landay, J. A. (2002). Where do web sites come from?: capturing and interacting with design history. *ACM CHI*. 1-8.

22. Kurlander, D. and Feiner, S. (1992). A history-based macro by example system. *ACM UIST*. 99-106.

23. Kurlander, D. a. F., S. (1988). Editable Graphical Histories. *IEEE Workshop on Visual Languages*. 416-423.

24. Kurlander, D. a. F. S. (1990). A Visual Language for Browsing, Undoing, and Redoing Graphical Interface Commands. *Visual Languages and Visual Programming*. 1990, Plenum Press. 257-275.

25. Lieberman, H. (1993). *Mondrian: a teachable graphical editor. Watch what I do: programming by demonstration*, MIT Press. 341-358.

26. Little, G., Lau, T. A., Cypher, A., Lin, J., Haber, E. M. and Kandogan, E. (2007). Koala: capture, share, automate, personalize business processes on the web. *ACM CHI*. 943-946.

27. Liu, X., Mei, T., Hua, X.-S., Yang, B. and Zhou, H.-Q. (2007). Video collage. *ACM Multimedia*. 461-462.

28. Meng, C., Yasue, M., Imamiya, A. and Mao, X. (1998). Visualizing Histories for Selective Undo and Redo. *Asian Pacific Computer and Human Interaction*. 459.

29. Minneman, S., Harrison, S., Janssen, B., Kurtenbach, G., Moran, T., Smith, I. and Melle, B. v. (1995). A confederation of tools for capturing and accessing collaborative activity. *ACM Multimedia*. 523-534.

30. Myers, B. A., McDaniel, R. G., Miller, R. C., Ferrency, A. S., Faulring, A., Kyle, B. D., Mickish, A., Klimovitski, A. and Doane, P. (1997). The Amulet Environment: New Models for Effective User Interface Software Development. *IEEE Transactions on Software Engineering*. 23(6):347-365.

31. Myers, B. A., Weitzman, D. A., Ko, A. J. and Chau, D. H. (2006). Answering why and why not questions in user interfaces. *ACM CHI*. 397-406.

32. Mynatt, E., Igarashi, T., Edwards, W. and LaMarca, A. (1999). Flatland: New dimensions in office whiteboards. *ACM CHI*. 346-353.

33. Nakamura, T. and Igarashi, T. (2008). An application-independent system for visualizing user operation history. *ACM UIST*. 23-32.

34. Plaisant, C., Milash, B., Rose, A., Widoff, S. and Shneiderman, B. (1996). LifeLines: visualizing personal histories. *ACM CHI*. 221-227.

35. Plaisant, C., Rose, A., Rubloff, G., Salter, R. and Shneiderman, B. (1999). The design of history mechanisms and their use in collaborative educational simulations. *ACM CSCW*. 44.

36. Ramos, G. and Balakrishnan, R. (2003). Fluid interaction techniques for the control and annotation of digital video. *ACM UIST*. 105-114.

37. Rekimoto, J. (1999). Time-machine computing: a time-centric approach for the information environment. *ACM UIST*. 45-54.

38. Shipman, F. and Hsieh, H. (2000). Navigable History: A Reader's View of Writer's Time. *New Review of Hypermedia and Multimedia*. 6:147-167.

39. Su, S. (2007). Visualizing, Editing, and Inferring Structure in 2D Graphics. *UIST Doctoral Symposium*.

40. Terry, M., Bunt, A., Kay, M. and Lafreniere, B. (2009). ingimp: A Smorgasbord of Usability, Adaptive UIs, and Visually Arresting Graphics. *Libre Graphics Meeting*.

41. Truong, B. T. and Venkatesh, S. (2007). Video abstraction: A systematic review and classification. *ACM Transactions on Multimedia Computing, Communications and Applications*. 3(1):3.

42. Yeh, T., Chang, T.-H. and Miller, R. C. (2009). Sikuli: using GUI screenshots for search and automation. *ACM UIST*. 183-192.