Chapter 2

# Interactive Usability Evaluation

Besides conventional features such as performance and robustness, usability is now recognized as an important quality attribute in software development. Traditionally, usability is investigated in controlled laboratory conditions, by recruiting a (hopefully representative) user sample and often performing video recordings and surveys that are later reviewed. This requires an important investment in time and money, not to mention that processing user interaction data is, at a minimum, cumbersome.

This chapter discusses the role of implicit interactions when performing usability tests on websites; concretely, *a)* which kind of data can be gathered by observing the overt behavior of users, without relying on explicit feedback, *b)* how this data can be presented to the usability evaluator, and *c)* which questions can be answered by inspecting such data.

## Chapter Outline

## 2.1 Introduction

Determining how UIs are operated has aroused historically a lot of interest in many research fields such as product design and software engineering. For instance, detecting areas of interest or misused layout spaces, time to complete a task, etc. In a typical usability evaluation study, it is important for practitioners to record what was observed, in addition to why such behavior occurred, and modify the application according to the results, if needed. Observing the overt behavior of users provides useful information to investigate usability problems. Based on live observations, or analyses of video tapes, an evaluator constructs a problem list from the difficulties the users have accomplishing the tasks [Jacobsen et al., 1998]. However, video data is time-consuming to process by human beings [Daniel and Chen, 2003]. Analyzing video has traditionally involved a human-intensive procedure of recruiting users and observing their activity in a controlled lab environment. Such an approach is known to be costly (e.g., equipment, personnel, etc.) and rapid prototyping sometimes requires just preliminary studies. What is more, software applications usually have a life cycle extending well beyond the first release. Problems like these have led to consider alternate approaches. Concretely, in the field of web applications, remote activity tracking systems are today one of the main sources to evaluate the UI and analyze user behavior.

Processing user interaction data is thus, at a minimum, cumbersome. Fortunately, today there is a vast array of tools that can facilitate this task to the researcher. For instance, state-of-the-art usability systems employ client-side logging software, which include mouse and keyboard tracking, since these input devices are ubiquitous; therefore neither specific hardware nor special settings are required to collect interaction data remotely. The rationale that justifies these remote logging methods lies on the fact that there is a strong correlation to how likely a user will look at web pages [Chen et al., 2001; Huang et al., 2012; Mueller and Lockerd, 2001], and hence a mouse can tell us the user's intent and interests most of the time.

Modern cursor tracking systems usually support replaying the user interactions in the form of mouse tracks, a video-like visualization scheme, to allow researchers to easily inspect what is going on behind such interactions; e.g., *How many of the users did actually click on the "Buy" button? In which order did the user fill in the form fields? Do users ever scroll the web page? If so, how far exactly?* Nonetheless, traditional online video inspection has not benefited from the full capabilities of hypermedia and interactive techniques. We believe that mixing both channels is likely to better assist the usability practitioner. Therefore, our proposal is enhancing *hypervideo* technology to build a useful inspection tool for web tracking. Section 2.3 describes extensively the proposed system.

### 2.1.1 Lowering Usability Costs

Assessing the allocation of visual attention with conventional methods like click analysis, questionnaires, or simply by asking subjects where they have paid attention to, are limited to those processes which are part of conscious reflection and alive control. Relying exclusively on such methods will lead to a major validity problem, because attentional processes do not solely depend on user awareness. They are often driven beyond such awareness, and therefore are not reportable [Schiessl et al., 2003].

The eye movement is available as an indication of the user's goal before she could actuate any other input device [Jacob and Karn, 2003]. Unfortunately, an eye tracker is a very expensive hardware that requires exceptional calibration and needs to be operated in a laboratory with a small user sample, being not accessible to everyone [Nielsen, 2004]. Also, it has been shown that observers do not necessarily attend to what they are looking at and they do not necessarily look at what they are attending to [Toet, 2006]. On the contrary, measuring cursor activity is cheaper and quite affordable, since it does not require additional hardware, and enables remote data collecting. Moreover, in modern UIs, pointing devices such as pens, mice, trackpoints and touchpads, are ubiquitous [Ruiz et al., 2008]. Where there is a web browser, there is a mouse cursor [Chen et al., 2001].

Cursor tracking offers a series of interesting advantages when compared to traditional usability tools. According to Arroyo et al. [2006]: *1)* It can be mass deployed, allowing for large datasets. *2)* It is able to reach typical users and first time visitors in their natural environment. *3)* It can continuously test live sites, offering insight information as new content is deployed. *4)* And most importantly, it is transparent to the users, so no experimenter bias or novelty effects are introduced, allowing users to navigate as they would normally do. One can argue that mouse movements are noisy, but also eye movements—actually even when looking at a point. Furthermore, the eye has higher error rate than the mouse, i.e., the coordinates reported by an eye tracker are often less accurate than those reported by most manual input devices. Finally, an eye tacker is an always-on device (which leads to the *Midas Touch* problem[1]), so distinguishing between intentional selection and simple inspection is more challenging with eye-gaze based devices.

## 2.2 Related Work

Automatic recording of user behavior within a system (also known as instrumentation) to develop and test theories has a rich history in psychology and UI design. One methodology that has recently begun to show promise within the HCI field is automated tracking or event logging to better understand user

[1]Eyes are never "off", so every gaze has the potential to activate an unintended command.

behavior. While users are interacting with an application, the system logs all UI events in the background. This event logging strategy enables the usability practitioner to automatically record specific behaviors and compute traditional usability metrics of interest (e.g., time to completion, UI errors, and so on) more accurately. Without these tools, these measurements would require researchers to meticulously hand-code behaviors of interest [Kim et al., 2008].

Mueller and Lockerd [2001] set a precedent in client-side tracking, presenting preliminary research on mouse behavior trends and user modeling. Arroyo et al. [2006] introduced the concept of collaborative filtering (that is, working with aggregated users' data), and the idea of using a web-based proxy to track external websites. Finally, Atterer et al. [2006] developed an advanced HTTP proxy that tracked the user's every move, being able to map mouse coordinates to DOM elements. Beyond the usefulness of these systems, only Atterer et al. [2006] could track complex Ajax websites, and visualization was solely the primary focus of Arroyo et al. [2006], although it was limited to an image overlaid on top the HTML pages. We argue that incorporating time-related information may enhance human interaction understanding, to replay exactly how users interact on a website. For instance, hesitations on a text paragraph may indicate interest about that content; or moving the mouse straight to a link of interest would show familiarity with the page. To this end, this is where video capabilities come into play, which, to some extent, have been lately implemented in industry systems.

Amongst the popular commercial systems at present, ClickTale[2], UserFly[3], and LuckyOrange[4] are deeply oriented to web analytics, with limited support for (non-interactive) visualizations. On the other hand, Mpathy[5] and Clixpy[6] are more visualization centered, but they use Flash sockets to transmit data, and so they only would work for users having the Flash plugin installed. Therefore, depending on the target audience of the website, it could lead to missing a huge fraction of the visitors that could provide valuable insights about their browsing experience. Finally, other approaches for visualizing user's activity are DOM based (Tag tracker[7]), or heatmap based (CrazyEgg[8]).

Basically, commercial systems work as "hosted solutions", i.e., a software-as-a-service delivery model. These systems require the webmaster to insert a tracking script in the pages to be targeted. Then such a tracking script transmits the data back to the commercial server(s). Eventually, registered users can review the tracking logs at an administration area or "admin site" provided by the commercial system.

[2] http://clicktale.com
[3] http://userfly.com
[4] http://luckyorange.com
[5] http://m-pathy.com
[6] http://clixpy.com
[7] http://otterplus.com/mps
[8] http://crazyegg.com

## 2.3 Simple Mouse Tracking

Having looked at the literature, there are still some niches that are not fully covered by current tools. Mainly, there is no possibility to visualize the behavior of simultaneous users at the same time, and no system does report metrics related to user-centered data. These facts motivated the development of a new tool which, besides incorporating most of the state-of-the-art features, differs significantly from previous work, as stated in the next section. Now we shall describe SMT2 [Leiva and Vivó, 2012], our previous work, and how it differs from current systems. Then, we introduce a new version, SMT2$\epsilon$, and show how it differs specifically from SMT2. Our tool is released as open source software, and can be downloaded and inspected at `http://smt2.googlecode.com`.

### 2.3.1 Overview of smt2

First of all, an important feature of our previous work regarding to state-of-the-art web tracking systems is the ability of compositing multiple interaction logs into a single hypervideo. This feature has been proved to be useful in assessing qualitatively the usability of websites, and also to discover common usage patterns by simply inspecting the visualizations (see Section 2.4).

Secondly, another important feature of SMT2 is the generation of user and page models based on the automatic analysis of collected logs. In this regard, we did not find any related tracking system that would perform implicit feature extraction from users' interaction data; i.e., interaction metrics inherently encoded in cursor trajectories. We believe that this is a promising line of research, and currently is gaining attention from other authors; e.g., Guo and Agichtein [2010]; Huang et al. [2011].

Thirdly, the recording approach used in SMT2 is different regarding the ones described in current industry systems. Concretely, we perform a discretization in time of user interactions, following a simple event logging strategy together with the *polling* technique; i.e., taking a snapshot of the cursor status (mainly coordinates, clicks, and interacted elements) at a regular interval rate. This way, SMT2 tracks the user actions as they were exactly performed, allowing also to modify the speed at which movies can be replayed.

### 2.3.2 Introducing smt2$\epsilon$

Regarding tracking capabilities, SMT2$\epsilon$ behaves almost identically as its predecessor, with the notable exception that SMT2$\epsilon$ features LZW compression to transmit the logged data, saving thus bandwidth. The actual improvements made to SMT2 that eventually derived in SMT2$\epsilon$ are focused on the server side.

To begin, our current effort goes toward interactive hypervideo synthesis from user browsing behavior. However, unlike conventional hypervideo, SMT2$\epsilon$ is aimed to build full interactive movies from remotely logged data. Furthermore, current hypervideo technology itself is limited to clickable anchors [Smith and Stotts, 2002]. SMT2 augmented this technology with *interactive* **infographics**, i.e., a series of information layers that are rendered at runtime and provide the viewer with additional information. For instance, hovering over a click mark displays a tooltip showing the cursor coordinates, or hovering over a hesitation mark displays the amount of time the cursor was motionless.

SMT2$\epsilon$ extends this hypervideo technology with: 1. **hyperfragments**: videos can be linked to specific start/end parts, and 2. **hypernotes**: HTML-based annotations that point to specific video parts. These novel improvements are convenient in a tracking visualization scenario for a series of reasons. First, hyperfragments allow the viewer to select a portion of the video that may be of particular interest. Hyperfragments can be specified either with a starting or an ending timecode. This lets viewers quickly access desired information without having to watch the entire replay. Second, hypernotes allow the viewer to comment on the video at a specific point in time; e.g., to point out some video details or to let co-workers know that such video has been reviewed. When a hypernote is created, the viewer can click later on a note icon on the timeline that will seek the replay to the time indicated by the hypernote (Figure 2.3a). This provides viewers with indexing capabilities that can be extended to content searching. Fourth, the content of hypernotes is HTML, which enables rich-formatted text and insertion of links and images. This capability opens a new door to how visualizations can be later processed; e.g., it would be feasible to build narratives that summarize a user session.

In addition, SMT2$\epsilon$ features two installation modes: as an all-in-one solution (when website and admin site are both placed in the same server) and as a hosted service (website and admin site are both placed in different servers). SMT2 was limited in this regard, since to allow cross-domain communication, every website would require at least PHP support to forward the requests to the storage server (i.e., the admin site). With SMT2$\epsilon$, however, the only requirement for a website to be tracked is inserting a single line of JavaScript code, as other commercial systems do, so potentially any website can use it.

Finally, SMT2$\epsilon$ features page classification according to user behavior in real time, by automatically mining the generated user and page models. The inclusion of this functionality was motivated by the fact that the viewer may find it useful to discover common interaction profiles as well as to easily identify outliers [Leiva, 2011] as new users access the website.

### 2.3.3 Architecture

As described below, SMT2$\epsilon$ is composed of three fundamental parts: recording, management, and visualization. On the server side, any web server (e.g., Apache, LightHTTPd, or IIS) supporting PHP and MySQL is able to run both the admin site and the visualization application. The technology used to create such an interactive movies is a mixture of PHP (to query the database), HTML (to overlay the tracking data on top of it), JavaScript (to prepare the aforementioned tracking data), and ActionScript (to build the hypervideos).

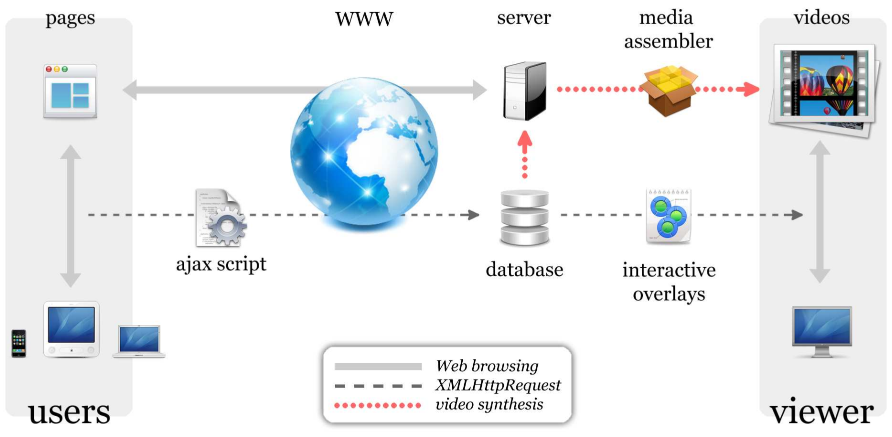


**Figure 2.1:** System architecture for acquiring users' activity and synthesizing interactive hypervideos.

### 2.3.4 Logging Users' Interactions

Every lower-level action can be recognized automatically, since the tracking script relies on the DOM event propagation model. We use the UNIPEN format [Guyon et al., 1994]—a popular scheme for handwriting data exchange and recognizer benchmarks—to store the mouse coordinates. This way, it is possible to re-compose the user activity in a reasonable fashion and to extract useful interaction semantics. While the user is browsing pages as she would normally do, an Ajax script logs the interaction data in the background. Tracking is performed in a transparent way for the users, either silently or by asking their consent.

It is worth pointing out that our strategy for transmitting the logged data do not rely on performing a server request each time a browser event is detected, as most tracking systems do. Instead, we store the data in a buffer, and we flush it at time-regular intervals. Doing so allows to reduce dramatically the number of HTTP requests to the web server, and hence lowering the overhead. Moreover, tracking can be *continuous* (default behavior) or *intermittent* (i.e., tracking stops/resumes on blur/focus events), letting the webmaster decide which operation mode is best suited to their needs. For instance, if an eye tracker is going to be used together with our system, then it is preferable to use continuous recording, in order to keep mouse and eye coordinate streams synchronized.

**Figure 2.2:** A working example of inserted tracking code. Here we set the registration frequency to 24 fps and establish a maximum recording timeout of 1 hour. We also set random sampling for user selection, and ask consent to the chosen users for monitoring their browsing activity (they must agree to start recording).

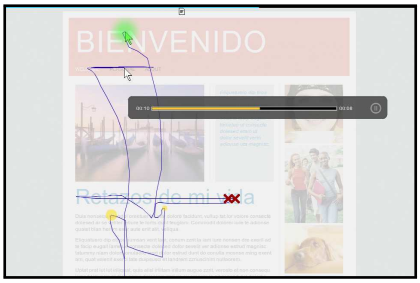
```
<script type="text/javascript">
smt2.record({
  fps:       24,
  recTime:   3600,
  disabled:  Math.round(Math.random()),
  warn:      true
});
</script>
```

On the contrary, if the system is used on its own then the webmaster may want to save storage space in the database by enabling intermittent recording.

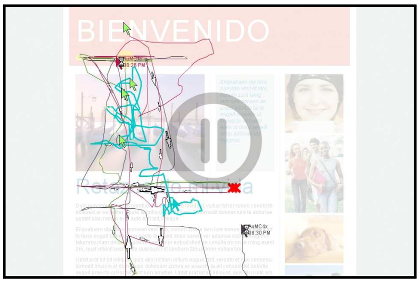
Another interesting logging feature is that the system can be invoked manually, if one have administrative rights to modify files in the web server, but it also can fetch external websites by using a PHP proxy that automatically inserts the required tracking code (Figure 2.2). We also take into account the user agent string to cache an exact copy of the page as it was originally requested, to avoid rendering differences due to different CSS being applied (e.g., on mobile devices compared to desktop computers). Additionally, it is possible to store interaction data from different domains in a single database, provided that each domain and the database are under the webmaster control.

### 2.3.5 Video Synthesis

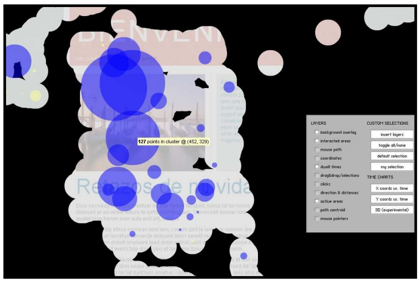
The process to create an interactive hypervideo is composed of four main tasks: *1)* mining, *2)* encoding, *3)* rendering, and *4)* event dispatching. First, we query the database with the information that the viewer provides. Creating this kind of movies by using web technologies allows adding interactive information to on-screen visualizations, ranging from basic to more advanced playbacks. For example, she might request to visualize a single browsing session. The system will then retrieve the subsequent logs to make a video that will replay all tracks sequentially. On the contrary, though, the viewer might want to filter logs by operating system and page URL, in which case she uses a data mining form. In this case, data are retrieved according to the indicated filtering options, and logs will be merged into a single hypervideo when replaying (Figure 2.3). Different mouse trajectories will be normalized according to the original viewport of the user's browser and the current viewport of the viewer's browser. The normalization consists of a non-uniform affine mapping (either by scaling or translating the coordinates, depending on the type of layout: namely *fixed*, *centered*, or *liquid*). Then, a cached copy of the browsed page and the above-mentioned interaction data are bundled in a hypermedia player. This way, movies can be replayed within any web browser.



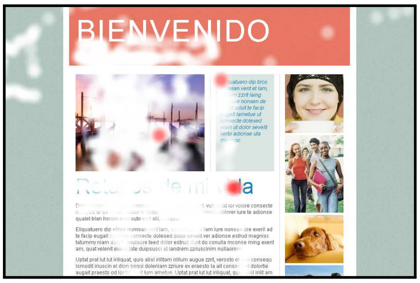
**(a)**



**(b)**



**(c)**



**(d)**

**Figure 2.3:** Some examples of our hypervideo visualization tool. [2.3a] single session with embedded media player. [2.3b] Replaying users' trails simultaneously, highlighting the average mouse track, and overlaying direction arrows. [2.3c] clusters of mouse movements, displaying also masked areas of activity. [2.3d] Dynamic heatmaps of mouse coordinates and clicks.

### 2.3.6 Interacting with the Data

On the server side, a multi-user admin site manages and delivers the hypervideos, allowing the viewer to customize a series of visualization options (Figure 2.3). The viewer can toggle different information layers interactively while she visualizes the videos by means of a control panel (Figure 2.4).

Automatic analysis of interaction features is also feasible for mining patterns within the admin site, since collected data are readily available in the database. This way, besides explicit metadata that is assigned to content, implicit knowledge can help to get a better picture on the nature of such content (see Section 2.5). Concretely, the metrics that SMT2$\epsilon$ computes for a given web page are described as follows.

**Time** Browsing time (in seconds) spent on the page.

**Clicks** Number of issued mouse clicks.

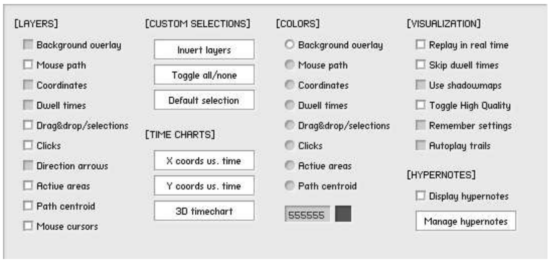


**Figure 2.4:** A draggable control panel is the main link between the viewer and the synthesized hypervideos. One can manipulate different visualization possibilities, which will be applied at runtime.

**Activity** Fraction of browsing time in which the cursor was moving, defined in $[0, 1]$. (0: no movements at all, 1: otherwise).

**Length** Cumulated sum (in px) of cursor distances.

**Distance** Average euclidean distance (in px) between coordinates.

**Entry/exit points** The first and last mouse coordinates, respectively.

**Centroid** Geometric center of all coordinates.

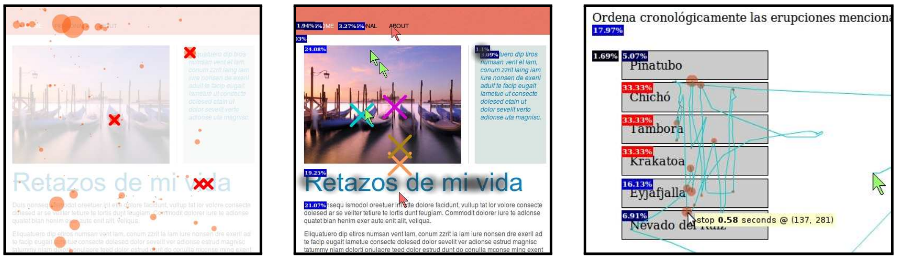
**Amplitude** Difference (in px) between maximum and minimum coordinates.

**Scroll reach** Percentage that informs how far did the user scrolled the page, defined in $[0, 1]$. (0: no scroll at all, 1: scroll reached the bottom of the page).

## 2.4 Applications

The following is a succinct list for illustrating the pragmatic utility of our system. We hope that the reader will be able to find other questions answered by examining other visualization marks.

- **Where do users hesitate? How much?** We followed the notion of *dwell time* introduced by Müller-Tomfelde [2007], i.e., the time span that people remain nearly motionless during pointing at objects. Dwell times are usually associated with ambiguous states of mind [Arroyo et al., 2006], possibly due to a thinking or cognitive learning process. In SMT2$\epsilon$ dwell times are displayed as circles with a radius proportional to the time in which the mouse does not move (Figure 2.5a). The system takes care of extremely large values of dwell times, by limiting the circle radii to a quarter of the viewport size.
- **Do users perform drag&drop operations? How?** Users perform drag and drop to select HTML content, or also to download an image to their desktop or to a file manager window. At a higher level, a web application



(a) (b) (c)

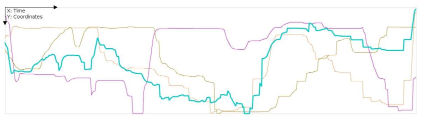
**Figure 2.5:** Combining visualization possibilities. [2.5a] Displaying hesitations (circles) and clicks (small crosses). [2.5b] Displaying entry/Exit coordinates (cursor bitmaps), motion centroids (big crosses), drag&drop activity (shaded fog), and interacted DOM elements. [2.5c] Analyzing a decision process; the user rearranged items in a list. Small circles represent dwell times. Hovered DOM elements are labeled based on frequency (percentage of browsing time), including a blue color gradient (100% blue: most hovered items). The same scheme is used to analyze clicked items, but using the red palette.

can support rearranging widgets to customize their layout, or also by adding objects to a list to be processed. Since we are using the UNIPEN format to encode each pair of mouse coordinates, the status of the click button can be easily represented, so SMT2$\epsilon$ provides a specific visualization type for these cases (e.g., Figure 2.5b).

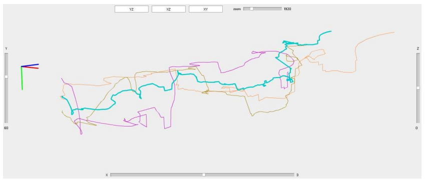
- **Which elements is the user actually interacting with?** Thanks to the bubbling phase of JavaScript events, whenever a mouse event is dispatched (e.g., `mousemove`, `mouseover`) the tracking script traverses the DOM hierarchy to find if there is an element that relates to the event. Each tracking log holds a list of interacted DOM elements, sorted by time frequency (Figure 2.5c), so such list can be inspected either quantitatively (by looking at the numbers) or qualitatively (by looking at the colors). This visualization can be helpful to answer related questions, such as if the users go straight to the content or whether the mouse hovered over a link without clicking.

- **Which areas of the page do concentrate most of the interaction?** To answer this question, a K-means clustering of the coordinates is performed each time a mouse track ends replaying. So, focusing on the clustered areas allows to visually notice where users are performing most of their actions. Each cluster is represented by a circle with a radius proportional to the cluster population (Figure 2.3c). This visualization layer is notably appropriate when tracking data are rendered as a static image.

- **Do different mouse tracks correlate?** The viewer can select the 'time charts' option from the control panel (Figure 2.4) and compare multiple tracks simultaneously (see Figure 2.6). The coordinates are normalized in



**(a)** Normalized coordinates against time



**(b)** Interactive 3D visualization

**Figure 2.6:** Time charts visualization. Bold line is the averaged mouse track, taking into account the selected users. The 3D view allows rotating the axes with 3 sliders (one for each direction), zooming, and projecting the lines in the YZ, XZ, and XY planes.

width and height according to the available chart size, to avoid possible visual biases.

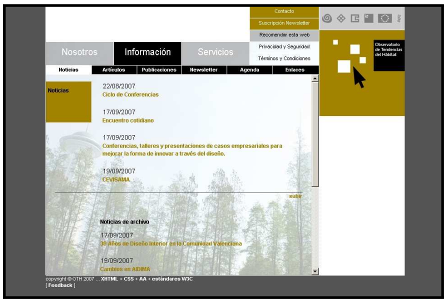
- **What is the persistence of the page?** In this case, a 3D visualization might be useful (Figure 2.6b). The 3D chart renders the evolution of each pair of cursor coordinates $x, y$ along the $z$ axis, and provides simple interactive controls to ease further inspection. This way, for a given page, the viewer can observe at a glance the duration of each visit and how do they relate to the rest of them.

## 2.5 A Case Study

Here we provide empirical evidence for the efficacy of SMT2$\epsilon$ as a usability inspection tool. To test the system in a real-world scenario, the system was presented to a team of five graphic designers that were not usability experts. They wanted to redesign a corporative website, and they all used the tool for one month. One of them assumed the super administrator role, and the rest of them were assigned to the admin group. Thus, everyone could access to all admin sections without several restrictions; e.g., the difference between a user in the admin group and the super administrator is that admin users neither can download nor delete tracking logs, create user roles, or dump the database from the admin site.

### 2.5.1 Qualitative Results

Designers ran an informal usability test on their own. They configured SMT2$\epsilon$ as indicated in Figure 2.2, and gathered a representative user sample (near 5000 logs) in two weeks. Potential problems could be identified when visually inspecting the hypervideos, either for single users or by aggregating the logs from commonly browsed pages. Designers noticed that some areas of the home layout were causing confusion to most users; e.g., people hesitated over the main menu until deciding to click a navigational item. Designers could also view that much of the interaction with the site was concentrated around the header section. Consequently, the team introduced some modifications to the web interface and gathered near 1000 logs in five days. This way, they could compare the generated interactions to previous data. Such updates had notable repercussions specially for first-time visitors (faster trajectories, less clicks overall). Figure 2.7 shows the appearance of the website before and after manually introducing the design updates. The reader can find more details of this study in [Leiva and Vivó, 2008].



Contacto
Suscripción Newsletter
Recomendar esta web
Privacidad y Seguridad
Términos y Condiciones

Nosotros | Información | Servicios

Noticias | Artículos | Publicaciones | Newsletter | Agenda | Enlaces

Observatorio de Tendencias del Hábitat

Noticias

22/08/2007
Ciclo de Conferencias

17/09/2007
Encuentro cotidiano

17/09/2007
Conferencias, talleres y presentaciones de casos empresariales para mejorar la forma de innovar a través del diseño.

19/09/2007
CEVISAMA

subir

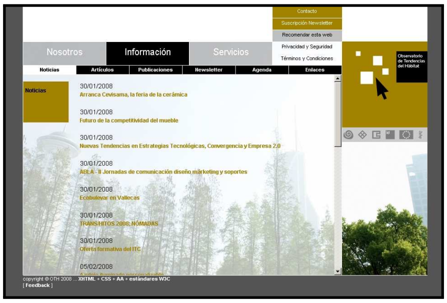
Noticias de archivo

17/09/2007
30 Años de Diseño Interior en la Comunidad Valenciana

19/09/2007
Cambios en AIDIMA

copyright © OTH 2007 ... XHTML + CSS + AA + estándares W3C
[ Feedback ]

**(a)**



Contacto
Suscripción Newsletter
Recomendar esta web
Privacidad y Seguridad
Términos y Condiciones

Nosotros | Información | Servicios

Noticias | Artículos | Publicaciones | Newsletter | Agenda | Enlaces

Observatorio de Tendencias del Hábitat

Noticias

30/01/2008
Arranca Cevisama, la feria de la cerámica

30/01/2008
Futuro de la competitividad del mueble

30/01/2008
Nuevas Tendencias en Estrategias Tecnológicas, Convergencia y Empresa 2.0

30/01/2008
ABLA - II Jornadas de comunicación diseño márketing y soportes

30/01/2008
Ecobulevar en Vallecas

30/01/2008
TRANS/HITOS 2008: NÓMADAS

30/01/2008
Oferta formativa del ITC

05/02/2008

copyright © OTH 2008 ... XHTML + CSS + AA + estándares W3C
[ Feedback ]

**(b)**

**Figure 2.7:** Website as it was designed initially (2.7a) and the redesigned layout (2.7b).

Overall, designers found the system very helpful. The main advantages suggested were being able to reproduce exactly what users did in a web page, and the speed with which a redesign could be verified. Concretely, the visualization layers (Figure 2.4) that the team found most useful were: mouse path, dwell times, clicks, direction & distances, and active areas. Designers also reported that there were two layers they found not relevant: path centroid and drag&drop/selections, mainly because *1)* the centroid was perceived as an imprecise indicator of the user interaction (i.e., designers stated that it was hard to derive meaningful conclusions by looking just at a single point on the screen) and *2)* only a few users performed drag&drop operations in the website. Designers liked the option of being able to switch to a static representation, specially when working with a large number of aggregated tracking logs.

## 2.5.2 Quantitative Results

Additionally, we asked permission to the team to download their gathered tracking logs for an offline study. They provided us with 4803 XML files. We processed them to build regression models of user activity and to create interaction profiles. We were able to predict with 71% of accuracy the expected time on a page based on the amount of mouse motion. Among other interesting findings, we noticed that the temporal evolution of mouse movements follows a log-linear curve. This showed up that, instead of the idiosyncratic distinction between active (exploratory) and passive (lurker) users, there exists a wide continuum of in-between behaviors. More details of the above mentioned experiment can be found in [Leiva and Vivó, 2008].

Additionally, we used cursor data for a behavioral clustering experiment, which is detailed in Chapter 3. Eventually, 95% over all browsed pages could be explained by looking at 3 meaningful profiles. Designers could then review the pages belonging to each profile, focusing on the identified behaviors, and could continue iterating over the design-develop-test process. Similar experiments on this behavioral clustering methodology can be found in [Buscher et al., 2012; Leiva, 2011].

### 2.5.3 Limitations

Web-based activity tracking systems have inherent limitations, and of course SMT2$\epsilon$ is no exception to this rule. Although measuring page-level interactions is cheaper and enables remote data collecting at large, the main drawback we have found is that assessing the allocation of visual attention based on interaction data alone is a non-trivial task. For instance, while it is commonly agreed that "a mouse cursor can tell us more" [Chen et al., 2001], Huang et al. [2011, 2012] have demonstrated that browsing time and user behavior have notable repercussions on gaze and mouse cursor alignment. Also, it has been shown that users do not necessarily attend to what they are looking at, and they do not necessarily look at what they are attending to [Toet, 2006]. Therefore, the usability practitioner should be aware of these facts before considering using a web tracking system, depending on the task that would be assessed or the context of their study.

On the other hand, our tool was designed to handle a limited number of simultaneous user sessions in the same hypervideo. One may note that if the system were used to show data from, say, 10000 concurrent users, then we believe the video visualization would not be much meaningful. Suffice to say it could be done, but at the cost of increasing the cognitive overload for the viewer (since visually inspecting too many users at the same time can be stressful), and only limited by the processing power of his computer. In this situation, aggregated data would work much better if rendered as a single image—discarding thus the temporal information but retaining interactivity for the viewer. This way, it is still possible to visually infer time-based properties such as mouse velocities (for instance, by looking at the 'directions & distances' layer, Figure 2.4).

Additionally, besides the fact that our tool normalizes the mouse coordinates to avoid possible visual biases while replaying the hypervideos, we noticed that sometimes the visualization is not perfectly accurate, partly due to JavaScript rounding errors, partly due to discrepancies between how browsers render CSS. These browser discrepancies can be greatly minimized by using a reset stylesheet on the web page. On the contrary, higher discrepancies are expected when the user access from a mobile device and the viewer uses a desktop computer. We are currently investigating different methods that would tackle this problem, which is common to all web-based tracking systems, and for which there is no trivial solution. For instance, the system could use the mobile user agent to fetch the page that the user visited, but it could happen that the page had changed since that visit, or even that it no longer exists. The same argument applies to the stylesheets of that page. Therefore, a more technically advanced approach should be taken into consideration, such as caching all assets for each user visits, at the cost of increasing the storage space.

## 2.6 Conclusions and Future Work

To better understand user behavior on the Web, modern tracking systems should rely on the browsing capabilities of the users, instead of the traditional server access logs. However, this approach has a clear trade-off, as moving to the client side involves having to process much more data. We believe that offering such data processed as a hypervideo can be considered as a promising idea and a specially helpful approach for assessing the usability of websites.

This article has described the design and implementation of SMT2$\epsilon$, a web-based system for automatically gathering, mining, selecting, and visualizing browsing data in an interactive hypermedia presentation, either as a video or as a static visualization. The tracking system collects fine-grained information about user behavior, and allows viewers to control what they watch, when, and *how*, by selecting diverse types of infographics.

We have reported the main differences between our tool and previous web tracking systems, including the state of the art and highlighting our contributions to the field. We have shown the value of enhancing video visualizations with interactive techniques to present the viewer with complex information quickly and clearly. We have also described a real-world usage scenario proving that our system is a feasible and realistic implementation.

Tracking page-level browsing activity with SMT2$\epsilon$ requires no real effort from the user, other than standard usage. It also requires no training and provides context for actions. Armed with this awareness, one may conduct both qualitative and quantitative studies, being able to complement existing methodologies on web browsing and human behavior. Therefore, we believe that SMT2$\epsilon$ is ready to extend its scope to a broader, interdisciplinary audience.

One of our priorities for future work is working on scalability and performance limits especially concerning high-recording speeds. We also plan to enrich the system with other types of behavior analysis, for instance working with eye-tracking data, as hinted in the previous section, since user interaction is inherently multimodal.

### 2.6.1 Some Notes on Privacy

Monitoring the user interactions at a fine-grained level can be very useful to help shaping a more usable website, or making it more appropriate to the behavior of their users. However, as in other web tracking applications, this work raises privacy concerns. We are interested in understanding web browsing behavior, but we also want the user to be respected, so we designed the SMT2$\epsilon$ system with that notion in mind.

First, we believe logging keystrokes could be employed for unfair purposes, depending on the uses that one could derive from this tool. For that reason, we rejected to log raw keystroke data and track only keyboard events instead, without registering the associated character codes. Second, we believe users should not be monitored without their consent. This is a webmaster's responsibility, but not doing so could be considered unethical in some countries. Therefore we recommend to ask always the user before tracking takes place. Furthermore, once a user has agreed to track, we advocate for asking her consent again after a prudential amount of time (e.g., a few hours, until the end of the browsing session, or when a tracking campaign finalizes). Third, we believe logged data should be stored in a server the webmaster owns, and not in one she cannot control. At least, it should be possible to let users access their (raw) data. We encourage commercial tracking systems to do so, since chances are there and current web technologies can support it. Finally, unlike most analytics packages that track other sites users have visited or the searches they have made, we do not collect other information than basic browser events derived from normal usage at the site where SMT2$\epsilon$ is included. This way, we try to avoid an illegitimate abuse of our system (e.g., without advising at all that users are being tracked or hijacking submitted form data). Above all, the ethical use of computers should be above any functionality or feature.

# Bibliography of Chapter 2

E. Arroyo, T. Selker, and W. Wei. Usability tool for analysis of web designs using mouse tracks. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*, pp. 484–489, 2006.

R. Atterer, M. Wnuk, and A. Schmidt. Knowing the user's every move – user activity tracking for website usability evaluation and implicit interaction. In *Proceedings of the 15th International Conference on World Wide Web (WWW)*, pp. 203–212, 2006.

G. Buscher, R. W. White, S. Dumais, and J. Huang. Large-scale analysis of individual and task differences in search result page examination strategies. In *Proceedings of the fifth ACM international conference on Web search and data mining (WSDM)*, pp. 373–382, 2012.

M.-C. Chen, J. R. Anderson, and M.-H. Sohn. What can a mouse cursor tell us more? correlation of eye/mouse movements on web browsing. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*, pp. 281–282, 2001.

G. Daniel and M. Chen. Video visualization. In *Proceedings of the 14th IEEE Visualization (VIS)*, pp. 409–416, 2003.

Q. Guo and E. Agichtein. Ready to buy or just browsing? detecting web searcher goals from interaction data. In *Proceedings of SIGIR*, pp. 130–137, 2010.

I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pp. 29–33, 1994.

J. Huang, R. W. White, and S. Dumais. No clicks, no problem: Using cursor movements to understand and improve search. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, pp. 1225–1234, 2011.

J. Huang, R. W. White, and G. Buscher. User see, user point: Gaze and cursor alignment in web search. In *Proceedings of the annual Conference on Human Factors in Computing Systems (CHI)*, pp. 1341–1350, 2012.

R. J. Jacob and K. S. Karn. *Eye Tracking in Human-Computer Interaction and Usability Research: Ready to Deliver the Promises*, chap. Section Comentary, pp. 573–605. Elsevier Science, 2003.

N. E. Jacobsen, M. Hertzum, and B. E. John. The evaluator effect in usability tests. In *CHI 98 conference summary on Human factors in computing systems*, pp. 255–256, 1998.

J. H. Kim, D. V. Gunn, E. Schuh, B. C. Phillips, R. J. Pagulayan, and D. Wixon. Tracking real-time user experience (TRUE): A comprehensive instrumentation solution for complex systems. In *Proceedings of the 26th annual SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 443–452, 2008.

L. A. Leiva. Mining the browsing context: Discovering interaction profiles via behavioral clustering. In *Adjunct Proceedings of the 19th conference on User Modeling, Adaptation, and Personalization (UMAP)*, pp. 31–33, 2011.

L. A. Leiva and R. Vivó. A gesture inference methodology for user evaluation based on mouse activity tracking. In *Proceedings of Interfaces and Human-Computer Interaction (IHCI)*, pp. 58–67, 2008.

L. A. Leiva and R. Vivó. Interactive hypervideo visualization for browsing behavior analysis. In *Proceedings of the 21st international conference companion on World Wide Web (WWW)*, pp. 381–384, 2012.

F. Mueller and A. Lockerd. Cheese: Tracking mouse movement activity on websites, a tool for user modeling. In *Proceedings of Extended Abstracts on Human Factors in Computing Systems (CHI EA)*, pp. 279–280, 2001.

C. Müller-Tomfelde. Dwell-based pointing in applications of human computer interaction. In *Proceedings of the IFIP Conference on Human-Computer Interaction (INTERACT)*, pp. 560–573, 2007.

J. Nielsen. Capturing thoughts, capturing minds? from think aloud to participatory analysis. Available at http://openarchive.cbs.dk/bitstream/handle/10398/6501/14-2004.pdf, 2004. Retrieved November 8, 2009.

J. Ruiz, D. Tausky, A. Bunt, E. Lank, and R. Mann. Analyzing the kinematics of bivariate pointing. In *Proceedings of Graphics Interface (GI)*, pp. 251–258, 2008.

M. Schiessl, S. Duda, A. Thölke, and R. Fischer. Eye tracking and its application in usability and media research. *MMI Interaktiv*, 6(1):41–50, 2003.

J. Smith and D. Stotts. An extensible object tracking architecture for hyperlinking in real-time and stored video streams. Tech. Report 02-017, Univ. North Caroline and Chapel Hill, 2002.

A. Toet. Gaze directed displays as an enabling technology for attention aware systems. *Computers in Human Behavior*, 22(4):615–647, 2006.