
CS 6763 Assignment 4

Comparative Web Design Tool Evaluation: Macromedia Dreamweaver MX vs. Mozilla Composer

December 9, 2004

Maria Cordell
mccordell@cc.gatech.edu

Markus Haas
mhaas@cc.gatech.edu

Andriy Naborskiyy
andriyn@cc.gatech.edu

Oriana Williams
gtg745u@mail.gatech.edu

ABSTRACT

This paper documents empirical research used to assess metrics of visual design tool quality for two current Web design programs, one commercially available, the other freely downloadable. We studied how users new to these products approached a common design task of adding a page to an existing site and compared the resulting page to the provided design standard. Results of the evaluation and specific conclusions are reported.

INTRODUCTION

Our evaluation looked at new user experiences with two visually oriented Web design tools: one commercial product, the other freely available for download. We wanted to know how the design experience varied across two different tools designed for very similar tasks. Are there significant differences, and if so, what are they? The two products differ in the extent to which they offer supporting Web design functionality, utilities, and extensions, but both should be equally capable in terms of common page creation and editing functions. But for new tool users attempting these common design tasks, where do the tools diverge? Does one product afford new users significant advantages for the given tasks? If so, in what way? With which product were new users more comfortable?

ANALYSIS PLAN

Our analysis plan evaluated how new users' interactions compared across two web design tools: Macromedia Dreamweaver MX and Mozilla Composer. Our original plan was outlined in [2]. The following plan discussion summarizes our methods and notes particular revisions to the original methodology.

Assessment Goals

We wanted to know how well each tool facilitated Web page creation for relative newcomers to visually oriented Web design tools. That is, we wanted to know how well each tool supported users who are relatively new to it, and what the visual design mode of each tool contributed to the Web page design/implementation process, the tool's approachability, and the overall quality of the experience. To best answer these questions and realize our assessment goals, we selected the following three interrelated metrics of tool quality :

- Learnability--the degree of user effort required to master tool functionality--lends itself well to an evaluation that focuses on users new to the design tool; the users already know what they want to do, but they must discover and use each tool's particular affordances to

achieve their task goals. In effect, learnability measured how well the design tools helped users transfer prior knowledge of HTML and Web page creation from a text-editing mode to a visually oriented Web design model.

- Functional completeness--how fully a tool supports the accomplishment of tasks in simple contexts--measured the degree to which each tool supported implementation of specific Web design elements. In our scenario, users were asked to perform a number of typical and basic tasks in HTML programming to convert a design into a working page. Including this metric allowed us to look for specific tool problems (bugs) or functionality omissions that our test subjects might encounter while attempting to complete their designated tasks.
- Pleasantness--which in this case we equate with lack of frustration--is a qualitative measure of tool usability; if a tool is difficult or frustrating to use it reflects poorly on the usability of the software. This can also point to issues of learnability and/or functional completeness. In our evaluation, lack of frustration is measured by the overall time required to complete the task and the apparent ease of use: few or no cases of "hunting" for a control or command.

Test Subject Criteria

We defined new users as those relatively new to visual Web design tools but with good understanding of at least basic HTML and the general requirements for creating a Web page. During the evaluation, each test subject would be asked to accomplish an unordered number of tasks by reapplying or transferring knowledge of HTML to the unfamiliar context of the visual Web design tools under evaluation.

To meet our assessment goals, we specifically sought test subjects who would approach the evaluation from a designer's perspective. We conducted our evaluation using six participants with varying backgrounds in HTML and Web page creation. Ideal candidates would have at least basic knowledge of HTML page composition and at most intermediate familiarity with either of the Web design tools under study.

Test Metrics

Each subject was asked to complete a set of unordered tasks in both Dreamweaver and Composer. We imposed a twenty-five minute time limit for each session and, to minimize ordering effects, switched the tool use order for half the subjects. Subjects were asked to add a third page to an existing two-page Web site design. The third page included dependencies on the existing pages and thus added a common yet important element of Web page design.

Specific techniques were invoked during the evaluation to compare our tool quality metrics. To evaluate functional completeness we focused on the user's ability to complete the given tasks. If properly done (assuming the tool is functionally complete in this sense), the final page should very closely match the supplied layout model (see Figure 1). This metric was evaluated post-test by examining the resulting page and comparing it to our standard. Questionnaire results (described later) provided insight to each user's perception of functional completeness for each tool.

Learnability was evaluated by observing how the user navigated tool options. The speed and apparent comfort with which previously explored controls were used was be subjectively evaluated through observation and by analyzing Likert scale questionnaire responses. Pleasantness was subjectively measured through similarly scaled questionnaire responses and provided quantitative results regarding the user experience.

Design Objective and Instructions

Our testing protocol focused on analyzing the larger building block components of creating a new Web page for an existing site. As in the case of the VLSI circuit layout task profiled in Chapter 10 of [1], our design objective enabled the Web page designer to generate tasks in any order.

Subjects were given a completed layout design to be implemented in each tool (see Figure 1). The subject's task was to convert the layout into a suitably and fully executed Web page. This approach may have bypassed the creative process of conceptualizing the page, but it nonetheless involved design in structuring the HTML page. The specific elements used to add content to a page and how that content is structured (using a table for gridding, for example) is an important aspect of Web design and can affect the maintainability and portability of a page. One common example of the latter surfaces in the way page appearance varies from one browser brand to another (e.g., Internet Explorer vs. Netscape) or even between different versions of the same browser.

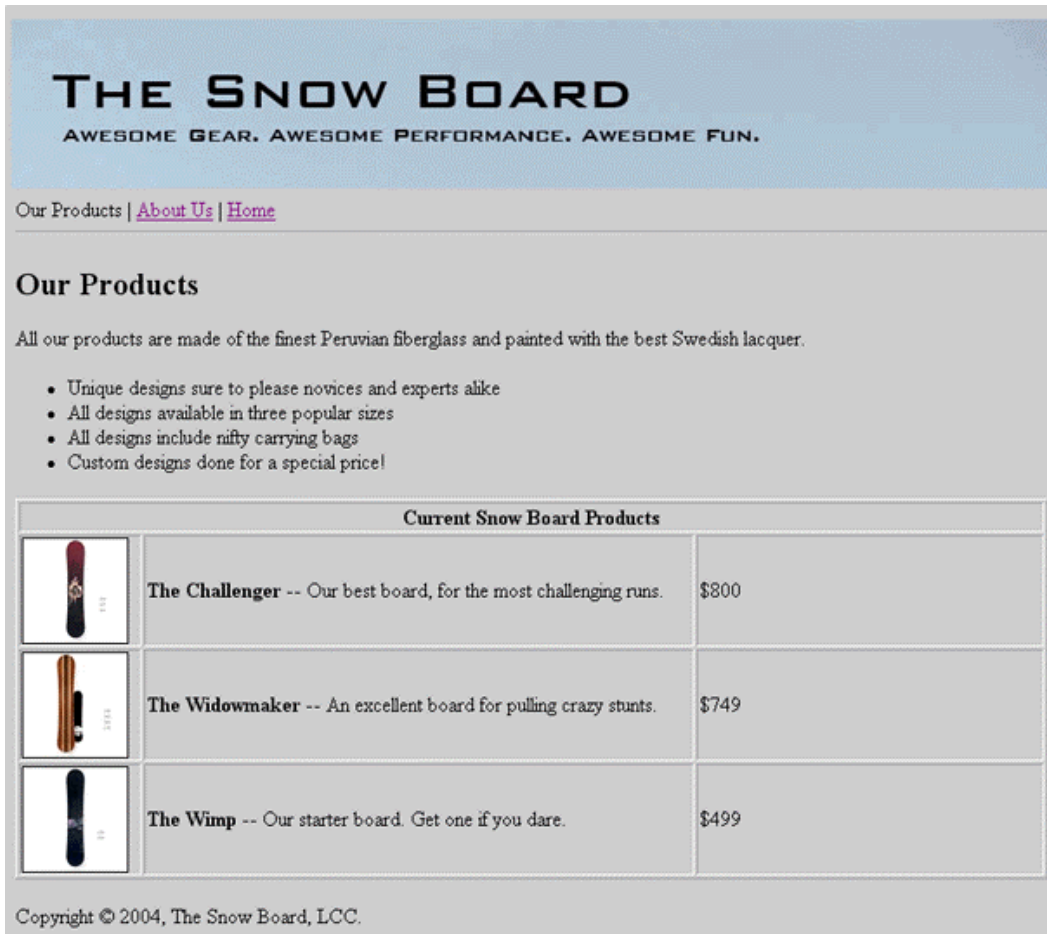


Figure 1: Users were provided with a hardcopy printout of the complete Web page layout. Each subject was asked to create this page using the tools under evaluation. Image files for the page banner (with the blue background) and products were supplied along with other site files.

The hardcopy of the page layout provides most of the information needed to execute the page layout in both design tools. Other required elements, including background color, page title, maximum content width, and image border, were specified in a separate instruction sheet. The instruction sheet also indicated the location of the site's working folder, where other site pages and all site images (including those needed for this page) were located.

Successfully creating the page required each subject to accomplish at least the following tasks in the selected design tool:

- create a new page from scratch or load an existing page as a starting point
- add or edit the page title
- change the default background color (if starting from a new page)
- add or edit text, including page heading elements, bulleted list, and a copyright symbol
- creating or editing the layout within provided page specifications
- creating or editing links
- creating or editing a multi-column table with a border and a merged-cell header row
- adding images to the page and to the appropriate table cells
- changing text attributes in the appropriate places (bolding text and/or heading sections)
- changing image attributes (adding an image border)

There are multiple approaches to implementing a design objective in HTML, and we expected some variation, not just between the two tools, but within each tool. The important aspect to capture was the specific tool functions selected to carry out each objective and how well the selected approach served the subject's needs. The results will directly inform each tool's assessment in terms of functional completeness.

Data Collection

Our data collection model was taken from sources [3] and [5] on design protocol analysis, which provide examples specific to design activity. We adapted our model to encompass the assessment goals listed above. Our process comprised passive subject observation, tool use session recording, and comparison of the resulting page design with our standard.

We used Camtasia, a software product for recording on-screen activity, to capture individual use sessions. We later played back the capture files to evaluate each session for our criteria according to a well-defined coding scheme. In particular we looked for any of the following events to occur. We recorded the event type as well as the time of the event, and times were rounded to the nearest half minute. Results are

- number of times help is accessed
- number of errors
- number of design omissions (at the end)
- number of times program help is consulted
- open existing page as guide
- time duration for completion of all tasks (overall session time)
- an "undo" operation
- idle browsing (hunting) periods
- number of times code (vs. design) mode used

In coding the sessions we recognized that there are two kinds of errors: tool errors (perhaps due to program bugs) and user errors. We also defined "undo" operations as actions done incorrectly then immediately corrected, with no intervening activity. Design omissions are tasks not done or incorrectly implemented tasks and were determined by examining the completed Web page. The following listed below were checked. Our findings are listed below in the Evaluation Results section.

- page background color & page title
- all page content within 800-pixel width maximum
- all images included; product images with 1-pixel border
- product table has border
- all hyperlinks correct
- bulleted product feature list
- horizontal rule
- page saved as "products.htm" within specified folder

We chose to have the subject answer four separate questionnaires--one before the test, one following completing each application's part of the experiment, and one after completing all other parts of the protocol. The rationale behind a pre-experiment questionnaire is to ask questions whose answers might be affected by the experience of doing the experiment's tasks. For instance, if the subject has a difficult time with the experimental task, he may discount his level of expertise at that type of task. An objective measurement of a subject's skill is valuable because it lets the researcher evaluate the difficulty of the experimental task. Other questions are not sensitive to being asked before or after the experiment, but are asked here to allow the subject to concentrate on recalling details of the just-completed tasks in the later questionnaires. These include standard questions characterizing the demographics of the subject group.

The intermediate questionnaires are given for two reasons. First, the experience of using the program is still very fresh in the subject's memory; one can easily forget or confuse details when the exact same task is repeated twice in a row, or more time has passed. Second, once the subject has used both tools, he can compare them and subjectively answer questions about one tool in a comparative manner with the other. For example, if the subject finds the first tool moderately easy to use, but finds the second one extremely easy to use, after completing both parts, the person is likely to assign a lower ease of use score simply based on the comparative ease of use of the second tool. The questions are identical for both tools.

The post-test is given to gain comparative data between Mozilla and Composer. The questions asked can serve two purposes- to gain additional insight into how well the subject was able to use the applications, and also to confirm information gathered in the screen capture. Another function of the post test is to allow the subject to comment on the experiment task. If he feels that his performance was negatively affected by a poorly worded task, or there were other factors that impacted his performance, that can be borne out here.

Detailed information including the questionnaires themselves can be found in [2], the seminal work in the field of proposals for comparative studies of Mozilla Composer and Dreamweaver.

EVALUATION RESULTS

Coding Scheme

Prior to coding all of the session captures, we went through a sample capture to develop and refine our planned event coding rubric. A standard rubric was allowed each evaluator to take notes on certain aspects regarding key user interactions that highlight the objectives of this study: to determine the learnability, functional completeness, and pleasantness goals surrounding both Macromedia Dreamweaver and Mozilla Composer. The following is the list of the coded actions with rationale for their importance:

Opening an existing page

- *Learnability*: By opening an existing page of the website, users have a better starting point than a if they just started with a blank page; an existing page gives them a view into particular representational modes for each application. For example, if a user inherits a table, mousing-over this table will give them important visual cues regarding its resizing, editing, and other editing modes. Hence, by opening an existing page, we hope to see if these novice tool users are able to more quickly gain familiarity with the scope of possible operations offered.
- *Coding Plan*: We documented whether a participant opened an existing page as a guide. We then observed to see if the potential learnability effects mentioned held true.

Idle Browsing or Hunting Periods

- *Learnability*: By documenting when participants begin to browse or hunt for an operation, we hoped to understand certain trends relating to learnability. To illustrate, if most of the initial browsing or hunting that takes place has a higher overall average time, while the idle

- browsing towards the end of the task is much shorter, then this suggests a certain level of the learnability of tool.
- *Pleasantness*: For the purposes of this paper, recall that pleasantness has been equated with lack of user frustration. Sometimes this frustration might be observed while other times it might be seen through the results of participants hunting expeditions. Are the participants able to achieve everything they wish through their menu and tool searches? Despite a tool's functional completeness, if participants' idle browsing results in fruitless searches, the product's pleasantness and ease of use can significantly decrease.
 - *Coding Plan*: To extract learnability and pleasantness conclusions from the session captures, we planned to observe the frequency and time range of each user's idle browsing in both the beginning and towards the end of their task. We also noted when idle browsing did not result in a successful search.

System Errors

- *Functional Completeness*: Although both Dreamweaver and Composer have been developed and improved for a number of years, they might still have errors, bugs, or inconsistencies that lead the participant to have approach tasks in roundabout ways or, in the worst case, prevent completion of the task. Serious errors of the latter type are typically unexpected, but possible, and directly relate to the functional completeness of the tool.
- *Coding Plan*: By watching the screen captures, we will code a tool error if the program responds in an unexpected way to a user-driven event. For example, we discovered early on in the evaluation that the horizontal rule in Mozilla Composer could not be successfully resized through the code view. We are curious to discover if any other system errors are bugs exist that compromise functional completeness.

User Errors leading to Design Omissions

- *Learnability*: Each subject's completed web page will likely have some range of dissimilarities with our standard. If so, document the decisions that led to these incorrect design choices. We wanted to know whether the correct action too difficult to find or whether the discrepancies resulted from users' inability to understand the functionality provided. Certain learnability aspects of the tool are in question if a user commits an error due to tool use difficulty.
- *Pleasantness*: If a user commits an error because they decided to give up a course of action in favor of another to get a similar result, this hints at an increased frustration level of the user.
- *Coding Plan*: Coding these type of user errors was difficult since we had to first note if the user attempted to apply the correct action to an object first. If an attempt was made but was unfruitful, then the user error might have been committed due to a weakness of the tool. On the other hand, we coded whether the user appeared to misinterpret the HTML object or action that needed to be taken. If so, the event was a misinterpretation error not due to the tool.

Undo operations

- *Learnability*: When action is applied to an object and then reversed, a participant has committed an "undo." If a subject performs an "undo," then we will document further "undo" actions committed on similar objects to observe if a user is committing the same mistakes more than once. If so, this implies some functionality within the tool might implicitly encourage user errors, and hence, the learnability curve relative to this function is too steep for novice users.
- *Coding Plan*: Since we planned to determine a pattern of tool learnability, we must code the relation of all of a user's "undo" actions that take place within their task. Recall that this does not just include using the undo command offered by the web design tools, but rather we documented every time a participant added/deleted/modified an object and then performed the opposite action to undo their previous action. In addition, we attempted to code each participant's ease or difficulty in recovery.

Help Access

- *Functional Completeness:* Our largest concern regarding the “Help” offered in both web design tools was the depth and breadth of its content or functional completeness. Here we were concerned with how easy Help was to find and to use. We also wanted to record whether Help contained the topics needed by our subjects, and whether the information located was useful for the task at hand.
- *Coding Plan:* The first plan to coding the functional completeness of Help was to determine the number of times that help is accessed. Secondly, the desired topic where was noted in addition to what subjects did with the information once it was located. We also noted whether, after accessing help, the participant was able to accomplish the questioned task or if the problem persisted.

Each of the coding events was recorded with the time of occurrence (to the nearest half minute). After the participant had completed the task, the number of design omissions (as compared with our standard) and the total time for the session was documented to provide more quantitative data with which to compare the two web design tools.

By analyzing the HTML file that resulted from individual sessions, we were able to determine whether or not each of the indicated design tasks were accomplished. We defined errors as omissions or incorrect implementation (e.g., a task was attempted but not properly or fully carried out).

Questionnaire Results

Each subject answered four questionnaires--one before any testing, one after using the each application, and one at the end. On the pre-test questionnaire, we characterized our subjects in terms of their fitness for our desired demographic, their experience with Dreamweaver and Composer, and their overall familiarity with composing web pages. From the questions in the opening survey, we learned that the subject population was close to our desired makeup. We sought a subject population that had a design-related major, and accomplished this in 5 out of 6 subjects. We also hoped that they would have at most moderate exposure to Dreamweaver and Mozilla Composer, but have some experience with designing web pages. The subjects were asked to rate their familiarity with Composer and Dreamweaver on a scale of 1 to 7, 1 being 'not familiar at all' and 7 being 'very familiar'. The average rating for Dreamweaver varied significantly at 3.0 +/- 2.0, and for Mozilla, was a 1.0 exactly. None of the subjects had ever used Mozilla composer, despite their use of Dreamweaver and the Macromedia Studio applications (3.8 +/- 1.8).

The questions pertaining to their comfort level in composing web pages through various means also showed a moderate level of experience, appropriate for our target demographic. When asked about their overall comfort level with creating a web page, the average subject response was a 4.67 out of 7, +/- 1.2, 1 being no experience on the Likert scale, and 7 being a great deal of experience. We broke the question down further to learn how they had done that HTML work. We asked whether the subjects had ever created a Weblog page, as the editing process is similar to a GUI HTML editor, although greatly simplified. Most respondents had no familiarity with weblogs, all giving a response of 1 except one 2. (1- no familiarity at all, 7- a great deal of familiarity) The final question on the opening questionnaire asked about the subjects' comfort with writing HTML code without a graphical editor. The responses here indicated an intermediate level of experience, 4.3 +/- 1.2.

This experiment featured a small subject population (n= 6), causing the rather high standard deviations. Nonetheless, we can show trends which are supported by the observations made from reviewing the screen captures of the user sessions.

The questionnaires given after the first and second applications (but before the final questionnaire) measured the subject's impression of the task just completed. The subjects were asked to rate the difficulty of the task, and how enjoyable it was to complete. For Dreamweaver, subjects rated the

difficulty as 3.0 +/- 0.9, while for Composer, it was a 3.3 +/- 0.8. (The scale here was again out of 7, with 1 being 'not difficult at all' and 7 being 'very difficult') When asked about the enjoyability of the task, Dreamweaver was rated as slightly more enjoyable, 4.33 +/- 0.8 versus 3.8 +/- 1.2. (1 being not 'enjoyable at all', and 7 being 'very enjoyable')

We sought to mitigate a learning effect by switching which product subjects used first, but due to sufficient differences in the interfaces, that effect did not manifest itself in these numbers. Difficulty for the first application was rated as a 3.17 +/- 1, and for the second one, 3.33 +/- 0.80. This is slightly opposite what one would expect for a learning effect. The enjoyability numbers likewise showed no effect, being identical at 4.17 +/- 0.8 for the first application, and 4.17 +/- 1.2 for the second. One subject, when giving general feedback, did indicate having an easier time with Dreamweaver because of first doing the task in Composer. That subject's completion time was lower in Composer, but otherwise, no corresponding effect was seen in any other participants.

Also in the intermediate questionnaires, the subjects were asked whether or not they were able to complete the task, and whether the tool provided all of the needed functionality. For both programs, 5 out of 6 subjects indicated that they were able to complete the task. However, when queried about if the necessary functionality was present, two subjects indicated that Composer did not, while all said that Dreamweaver did. (Being able to complete the task without having the necessary functionality in the tool is a bit difficult to understand, but it should be mentioned that this subject showed significant difficulty working with each program.) There is a rough correlation from this data to the total number of errors among all subjects per program. Mozilla subjects had 16 uncorrected errors on their final pages, whereas Dreamweaver subjects had only 12. Also possibly responsible for this high number (4 subjects contributed all the uncorrected errors) is the fact that not all users were able to complete the assigned task in the allotted 25 minutes. One ran out of time in Dreamweaver versus three in Composer. It is unclear why the subjects reported successfully finishing the task despite being stopped at the end of the available time. Finally, in response to the question of which tool they would choose to implement more complicated features on a web site, 5 out of 6 chose Dreamweaver. While the numerical data obtained from the Likert scale questions shows no significant difference in opinion about the programs, the objective measurements of errors made and time spent on doing the task does give Dreamweaver an apparent advantage in usability.

Also in the final questionnaire, the subjects are asked two questions concerning the task they worked on. The purpose for asking these was to give the subject a chance to indicate any problems or confusion that arose out of the way that the task was given. The first question asks if the written part of the task was clearly worded. The overwhelming response was that it was (6.8 out of 7, +/- 0.4, with the anchors of the Likert scale being 'not clearly at all' and 'very clearly'. The second question refers to the web page image that they were given. It asks "Did the illustration of the page you were to create provide all of the information necessary to create it?". Again, the response was very positive, a 6.67 out of 7, +/- 0.5.

Question	Familiarity (1 – 7 scale)
How familiar are you with Dreamweaver?	3
How familiar are you with the applications in the Macromedia Studio suite?	3.83
How familiar are you with Mozilla composer?	1
How much experience do you have with creating a web page?	4.67
How much experience do you have with creating Weblog pages?	1.17
How comfortable are you in writing HTML code (with no graphical editor or tools)?	4.33

Macromedia Dreamweaver MX Results

Total Session Time

The time each participant took to achieve the task to complete the task was measured not just by the length of the screen capture video, but rather when mouse movements began on the screen to the point in which mouse movements of the participants significantly decreased. Each participant's total time has not been adjusted to accommodate for the fact that they were not constantly working on the project. To clarify, many participants took breaks to read the instructions and inspect the expected web page. This behavior was observed and supported by the lack of mouse seen in the screen capture video. Table 1 below documents Dreamweaver time for each subject.

Use of Help

Within the six Dreamweaver sessions, Help was only accessed twice by the evaluation participant, subject 2. Prior to opening help, subject 2 was attempting to change an attribute of her table. When she began opening help she first chose "Extension Help", which describes help features associated with Dreamweaver's extensions, and searched on the word "tables". Subject 2 soon realized it was the wrong type of help for her needs, so she searched the Help menu item again, this time choosing "Using Dreamweaver" with hesitation. When the "Using Dreamweaver" help did open, she no longer entered a string to searched, but rather, just scanned the topics offered in the index. She selected and scanned the section on "Fonts/Encoding" then exited help and continued working on editing her table.

Design Omission or Incorrectly Executed Task Errors

Certain user errors occurred within the tool evaluations that lead to incorrect end products. One participant, subject 6, misinterpreted that a header image existed. Instead, she created a similar looking structure by using a table filled with color and similar text as the header image contained. Subject 2 also made a few misinterpretation errors. For example, instead of using an image border behind the pictures within the table, she applied a black background color to the cells containing these images. She chose to bold all of the text on the entire page. Finally, she also left an artifact from the existing page she opened—the background image remained present in her completed implementation. Subject 3 also had a misinterpretation error leading to an error in her implementation. She resized the table so that it was beyond 800 pixels in width despite the direction's explicit instructions that the table was to be exactly 800 pixels in width.

Open Existing Page

Only two participants, subject2 and subject4, opened an existing page as a starting block for the page they needed to implement. Here they were able to inherit reusable page objects like, the image header, copyright symbol, background color and the table depending on the page they opened.

Undo Operations

The amount of obvious "undo" operations made by participants were minimal and were not, typically, repetitively made in similar situations. Attempting to change highlighted text into a hyperlink, subject 1 right clicked and selected "Insert Link." After filling out the fields describing the name and source of the link, subject 1 selected "Okay" which resulted in a link placed adjacent to his highlighted text. In short, subject 1 created a new link while he intended to turn his highlighted link into text. The next time he did this, he selected the highlighted text and right clicked to choose "Make Link" which worked as he expected the first time. Subject 1 also went through another "undo" phase when he created a table for the header image. He first filled in the color this table to be blue. After looking back at the picture, he realized that the header was an image he was supposed to insert, not create. In this case, he corrected his mistake by filling in the color the newly created table to be white instead. The next final undo was made by subject 2. After creating the table, she inserted in the table images and resized them to fit the default width of the cell that contained them. After looking back at the picture, she realized that she had made them

to wide so manually resized them back down. Since she resized them manually, the pictures lost some of their consistency and were noticeably no longer of the same size.

Idle Browsing or Hunting

Idle browsing or hunting is how we categorized users explored to find needed operators. This was usually indicated by scanning mouse movements that searched through menus or paused to see an icon's tool tips indicators. Only hunting periods of obvious lengths of ten seconds or more were documented.

If two or more participants searched for the same type operation, the resulting idle browsing will be noted. For example, two participants (subject 1 and subject 3) began their task through exploring. Exploring can be defined as searching through the tool, simply to become familiar with the menu structure and tool functionality options. The two participants that began by their task by exploring did so to different degrees. Subject 1 very methodically went through all of the menus, tool bars, and menu tips. Subject 3's exploring technique was a less intense scan of the system functions.

After creating the table, participants needed to merge the top row of cells into just one cell. This subtask was a source of idle browsing from anywhere between ten to thirty seconds for subject 1, subject 2, and subject 6. Searches by subject 1 and subject 6 were successful, while subject 2's first idle browsing search for cell merging was unsuccessful. Subject 2 later went back after completing another task to find out how to merge the table's cells.

Applying a 1-pixel border to a series of images was the next challenge that caused participants to hunt for system functionality. The session capture for both subject 1 and subject 6 showed that they had to search for how to apply a border to an image. It should also be noted that although subject 2 never had direct difficulty with this task, she never attempted to apply an image border either.

The last function that two or more participants had to search for was a way to insert copyright symbol, ©, at the bottom of the page. Three subjects had to browse the menus in search of the symbol. Both subject 1 and subject 6 eventually found it, while subject 3 gave up searching prior to finding it. Subject 2 and subject 4 never had to insert the copyright symbol since they inherited it from the page from which they started.

For people unfamiliar with Dreamweaver and the Macromedia suite of products, the learnability curve experienced was higher than those participants familiar with any tool of the suite. One subject mentioned that "it was harder to pick up just by looking at it" and that "I had trouble figuring out how to do some things at first since the layout is different from what I'm used to."

Mozilla Composer Results

Total Session Time

Half the subjects using Composer were stopped at 25:00 minutes, but the page was basically complete. The other users completed on average within 13 minutes. Table 2 below documents Composer time for each subject.

Use of Help

Composer allows users to customize the size and location of the help window so that it is comfortable for the user and doesn't obscure the working area of the screen. One can also customize help content (resizable layout, formatting) so that the window is small yet can still help the user to consult all the necessary information.

How people used help was very individual. Most of the subjects were able to succeed in creating the page in Composer without consulting the help system. Only one subject #6 was effectively using help. The rest of the subjects didn't access help at all or tried to use it once without success.

Subject #6 had the help window constantly on screen and resized the help window to suit her needs. For the most part, subject #6 was able to find needed information to complete a task; the only exception was in setting the vertical alignment of the table.

The other subjects made one attempt to use help, and all were unsuccessful for the same reason. The subjects entered a very generic word for search, received a long list of articles where they couldn't immediately find relevant information, and aborted the attempt to use help system.

Design Omission or Incorrectly Executed Task Errors

Among all the subjects the most difficult part was limiting the page content to fit within 800 pixels. Half of the subjects didn't accomplish this. In particular, five of the subjects could not limit the width of the horizontal line. The one subject whose page had the line width correctly executed inherited this by starting the session with an existing page. The other subjects who found a proper dialog and entered 800 pixels width still were not able to complete the task; Composer wouldn't change the width even though the setting was correctly entered.

One subject created a nested bullet list within the list, which is not asked by the task. The subject could not see the error because the inner list had been given (by the user) the same alignment as outer one. Composer failed identify list nesting to the user.

Multiple subjects had problems setting the page title, setting width of image border, and creating horizontal line.

Undo

Most of undo operations were related to a table. All subjects attempted unsuccessfully several times to resize the column using control points. Very often rows or columns were deleted or moved unpredictably when subjects explored the behavior of the table controls. Several subjects couldn't manage the size of the columns without errors, even after succeeding once. The control-handles of the table also presented problems for users. The behavior of those controls is difficult to learn. In addition they were often source of errors, and often some part or even the whole table was unintentionally deleted.

Setting vertical alignment in the table cell was another common reason for undo. At first subjects tried to set horizontal centering instead of vertical centering and had to reverse results.

Several subjects had trouble terminating the bullet list. When they pressed enter the bullet list would continue on the new line. They had to undo and look for other ways terminate the list.

Idle browsing/hunting

All subjects but one explored the complete tool at the very start of the experiment. They looked at all available menus and hovered over all controls that present tool-tip. The first exploration of the tool was not enough for any of the subjects. All had to hunt for operators during the exercise. Most of the subsequent hunting was more granular (subjects were opening single dialog and investigating all the controls; or all items in one submenu). Still all subjects had to spend considerable time browsing for the operator to set background color (or otherwise didn't accomplish this task at all; some subjects were using pre-existing page as template and didn't have to do this task). Also, text alignment inside the cell took all subjects a lot of search through trial-and-error.

Macromedia Dreamweaver vs. Mozilla Composer

Learnability

The total time taken for each participant to complete the task varied in Dreamweaver and Composer. This might have been due to the fact that four out of the six participants reported a higher level of comfort (greater than or equal to 5/7 on the Likert scale) with Macromedia products. On the other hand, no participants were familiar with Mozilla Composer. This is potentially significant data since some Macromedia products follow the same layout structure as

Macromedia Dreamweaver. This might have given these particular participants an advantage achieving the tasks within Dreamweaver leading to lower task completion times than in Composer. Hence total time taken to complete the task is not a good indicator of the difference of learnability between the two tools.

The undo operations performed by participants were good indicators of trial and error attempts to explore the tool. In Dreamweaver the undo operations performed were mostly isolated events meaning that participants did not knowingly commit the same errors to the same objects. This suggests that users learned from their mistakes since they did not make the same errors twice. In Composer, the most difficult part to learn is related to table manipulation and alignment settings. Recall that all subjects attempted to resize the table through the typical drop and drag WYSIWYG technique, but were unsuccessful. Also, most participants spent a large portion of their time attempting to manipulate the table; this might have led to the increased amount of participants that did not complete the task within Mozilla. The difficulties associated with table control in Mozilla suggest that the functionality is very non-intuitive and lead to a higher learning curve within Composer than in Dreamweaver regarding manipulating design components.

In both tools, many participants had a slight amount of difficulty terminating the bulleted list since they never exited the “bullet list” mode. The source of this confusion was a modal issue that arises in most text editor tools. Even though many participants have had experience with bulleted lists, they still had the artifact of an extra bullet that they had to delete. Hence the learnability in both tools regarding this area could be improved upon.

Another way to learn the functionalities of a tool is through idle browsing. In Dreamweaver, some participants began by exploring, but this did not result in a reduced number of idle browsing incidents later on in their task. What was significant was that these idle browsing episodes that occurred later on in the task process were of much shorter time intervals. This implies that just over a short amount of time, the learnability of Dreamweaver increased a noticeable amount. In Composer, most subjects also did extensive exploration at the very beginning of exercise probably because they were more unfamiliar with the tool. After this initial exploration, subsequent pauses were much shorter. In fact, a lot of unusual operators (like copyright sign insertion) were accessed by most of the subjects without any idle pause at all. Hence, both Dreamweaver and Composer have good facility to support learning over time within a tool.

Functional Completeness

By looking at the help feature offered by both Dreamweaver and Composer, functional completeness issues were discovered in both programs. For example, in Dreamweaver, a participant had trouble accessing help since it was referred to as “Using Dreamweaver” instead of the expected title “Dreamweaver Help” or even more simply “Help”. The labeling should have been clearer to conform to some of the expectations of the user. When the user finally did find enter Dreamweaver’s help utility, she closed it and gave up without performing a search. In Composer three participants attempted to open help, but only one received relevant information that helped in completing the task. In both tools, better context sensitive help would be an improvement over the current help utility. Currently in Dreamweaver the context-sensitive help is available, but it is not obvious and was overlooked by the users seeking help. For it to be more effective it should be more affordable. This way participants do not need to know how to index their problem and can get richer, more meaningful data out of the help tool.

Functional completeness was also measured in terms of system errors found in the web design tools. In Dreamweaver, no system errors were found. On the other hand, Composer has several issues of concern in the functional completeness area. First, subjects were not able to set the width of the horizontal rule bar that had to add for the task. The tool would accept the participant’s input to change the setting, but ignored it and never applied the change. In result, there is no way to set this limit other than entering code-view and changing the html code manually. Also, users were constantly trying to use table controls to resize the columns width which suggests that this is an expected behavior missing in the tool. Subjects were dragging control points, but instead of a

resizing effect, their columns and rows were deleted. The only way this setting resizing can be achieved is through setting up the properties of the columns through a menu dialog. This large gulf of interpretation among the user's WYSIWYG expectations and the way the system performs is a sign of functional incompleteness of the Composer.

User Pleasantness

User pleasantness, or the lack of frustration, was determined by both observation and the user's ability to complete the operation as they expected. In Composer, for example, one subject was trying to align text in the table cell vertically, yet after a few minutes of attempting to do it correctly, gave up. Eventually the user resorted to adding empty line to achieve similar effect. Many similar frustrations were seen relative to manipulating and formatting tables in Mozilla Composer. One participant was even heard giving out a grunt of distain when her table was deleted unexpectedly. No similar obvious sources of frustration were documented or observed in Dreamweaver.

Tasks in Dreamweaver MX	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	group errors
Background page color	√	X	√	√	√	√	1
Page title	√	√	√	X	√	X	2
Content within 800 pixel max	√	√	X	√	√	X	2
1-pixel border for product images	√	X	√	√	√	√	1
Product table has border	√	√	√	X	√	√	1
All hyperlinks correct	√	X	√	√	√	√	1
Bulleterd product feature list	√	√	√	√	√	√	0
Horizontal rule	√	√	X	√	√	√	1
All provided images included	√	√	√	√	√	X	1
New page correctly saved	√	X	√	√	√	X	2
Individual errors	0	4	2	2	0	4	--
Session time	15:30	25:00	17:00	09:00	10:00	18:00	

Table 1: Design task errors (omissions or incorrect implementation) for subjects using Macromedia Dreamweaver MX. Subject 2 was cut off at 25 minutes.

Tasks in Mozilla Composer	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	group errors
Background page color	√	X	√	√	√	√	1
Page title	√	√	X	√	√	X	2
Content within 800 pixel max	√	X	X	√	√	X	3
1-pixel border for product images	√	X	X	√	√	√	2
Product table has border	√	√	√	X	√	√	1
All hyperlinks correct	√	X	√	√	√	√	1
Bulleted product feature list	√	√	√	√	√	X	1
Horizontal rule	√	X	X	√	√	√	2
All provided images included	√	√	X	√	√	√	1
New page correctly saved	√	√	X	√	√	X	3
Individual errors	0	5	6	1	0	4	--
Session time	14:30	25:00	25:00	12:00	12:00	25:00	

Table 2: Design task errors (omissions or incorrect implementation) for subjects using Mozilla Composer. Subjects 2, 3, and 6 were cut off at 25 minutes.

CONCLUSIONS

Our evaluation considered how new users would approach creating a Web page in two separate visually oriented Web design tools. We wanted to know how the experience varied and how well the tools supported the users in terms of functional completeness, learnability, and pleasantness. We also wanted to know if there were there significant differences between the tools in these areas, and if so, to identify those differences.

In general, we learned that people will apply gained domain knowledge when approaching a new tool. Our user population has experience in word processing, and particularly with Microsoft Word, and was able to transfer this knowledge to these applications. For example, most major applications that support word processing features have a standard method for inserting special symbols. Insertion of the copyright symbol is a case in point. Almost all subjects that didn't inherit the symbol (as the result of editing an existing page) knew how to find it on the first try because it was located where expected. This helped tool learnability, since users didn't have to spend time browsing for the symbol, and the symbol was inserted as expected.

Also with respect to learnability, we found that the look and feel of Composer led to relatively better subject performance for those parts of the tasks that were text- and formatting-oriented. For Dreamweaver there was a clear advantage in carrying out more Web page design-specific tasks, such as setting an image border. Although the task completion tables don't show a significant difference, review of the Camtasia captures showed Dreamweaver enabled users to more easily grasp the implementation of more advanced elements needed to complete tasks.

Text editing and text formatting actions were generally easier for subjects within Composer, at least initially. By default, Composer has a simpler and more familiar look, with a tool bar that lists common actions, especially for text formatting. Subjects displayed very little hunting when completing text-formatting task elements, indicating an immediate familiarity with the interface's affordances, even in cases where the subject was not familiar with the tool at all.

A significant difference in functional completeness surfaced in table editing. The way visual table editing works in Mozilla in is nonstandard for modern graphical design interfaces. Table editing

controls appeared to provide functionality needed for resizing but in fact had a different, unexpected result. Most subjects repeatedly tried to adjust table properties and attempted a variety of approaches to no avail. Lack of functional completeness also surfaced in Composer when users correctly set the width of the horizontal rule, but a bug prevented the setting from taking effect.

Repeated failures of this type led to subjects expressing frustration with the tool. (The questionnaire results show identical ratings for the pleasantness of using the tools, however, investigator observations confirmed user frustration.) Also, because of the amount of time spent on some task elements because of the above mentioned complications, three users were unable to complete the task in Composer (versus one user in Dreamweaver).

Overall, we learned that Dreamweaver is a better tool as the complexity of the task increases. Five out of six subjects said they would consider using Dreamweaver for future, more advanced work. The fact that Dreamweaver is perceived as having extensive functionality beyond that necessary for this experiment did not reduce its learnability. We couldn't accurately assess learnability over time (which we perceive to be a distinct metric) because our task was not sufficiently comprehensive and did not require repetitive behavior that would exhibit learning. But the frustrations shown by Composer users do indicate some design environment issues that should be addressed in future iterations of the product.

REFERENCES

1. Card, S. Moran, T., and Newell, A. *The Psychology of Human-Computer Interaction*. Lawrence Earlbaum Associates, 1983.
2. Cordell, M., Haas, M., Naborskiyy, A., & Williams, O. Assignment 3: Design Tool Analysis Plan. 2004.
3. Eastman, C (1970) On the analysis of intuitive design processes, G. Moore, ed. *Emerging Techniques in Environmental Design and Planning*, MIT Press.
4. Faraday, P. & Sutcliff, A. Providing Advice for Multimedia Designers. *ACM CHI 98*, 124-131, 1998.
5. Suwa, M. and Tversky, B. *What Do Architects and Students Perceive in their Design Sketches?* Design Studies.