A Usability Analysis of the Autopsy Forensic Browser

D.J. Bennett and P. Stephens

Department of Computing (Academic), Canterbury Christ Church University,
Canterbury, United Kingdom
e-mail: {david.bennett, paul.stephens}@canterbury.ac.uk

Abstract

This paper reviews the usability of the Autopsy Forensic Browser. Two expert-based usability review techniques are used: Cognitive Walkthrough and Heuristic Evaluation. The results of the evaluation indicate that there are many areas where usability could be improved and these are classified into areas of eight overlapping areas. Examples from each area are presented, with suggestions as to methods to alleviate them. The paper concludes with a future work proposal to effect the changes suggested and retest the current and the replacement system with a user-based evaluation.

Keywords

Usability, Human-Computer Interaction, Interface Design, Autopsy Forensic Browser, The Sleuth Kit, Cognitive Walkthrough, Heuristic Evaluation

1. Introduction

This paper reviews the usability of the Autopsy Forensic Browser tool. The reasoning for this is to improve future versions of the tool. In many ways forensic analysis tools are no different to any other mission critical software in terms of the need for usability. However, failure in usability in such tools could lead to outcomes which deny liberty to innocent persons, or enable criminals to continue with criminal activity should an investigation fail in some way.

1.1. Forensics and the Autopsy Forensic Browser

Gottschalk et al. (2005) define computer forensics as the identification, preservation, investigation, and documentation of computer systems data used in criminal activity. Tools to carry out these tasks are widely available commercially in the form of, for example, Guidance Software's Encase, AccessData's FTK, and ASR Data's SMART. Many of these tools are proprietary and therefore advocate a closed-source approach. Carrier (2003) argues that as these tools can dramatically affect people's lives by demonstrating innocence or guilt, they should be subject to a more stringent process of review than is available if the code of such systems is not published. Instead he promotes an open-source approach whereby the processes and procedures used are clearly defined and subjected to systematic review and debate. In this spirit he has created The Sleuth Kit and The Autopsy Forensic Browser.

The Sleuth Kit (sleuthkit.org, 2007b) is a set of command-line tools that allow an investigator to carry out an examination of a suspect hard-disk drive. It supports a number of disk and partition types, formatted by different operating systems, as well as a range of file systems. It is written in C and Perl and is based, in part, on The Coroner's Toolkit (Farmer & Venema, 1999). Although it usually runs on UNIX and Linux systems, it can be used on Windows utilising the Cygwin library (Cygwin, 2007; Lucas, 2004). The Sleuth Kit gives the investigator a high-level of flexibility and power when carrying out a digital investigation, however, this approach has disadvantages. The investigator needs to be an expert in UNIX-like commands and at least one scripting language. Without these skills examination of a complete suspect system would be a difficult and laborious task. With this in mind Carrier has created The Autopsy Forensic Toolkit (sleuthkit.org, 2007a). Autopsy is a web-based GUI to the command-line tools of The Sleuth Kit written using the Perl programming language. It is intended to make the collection and examination of digital evidence easier by automating many of the tasks. This includes case management, image integrity, keyword searching, preview and undeletion of files.

2. What is Usability?

There are several definitions of what is meant by the term usability, all based around the same base concepts. ISO9241 defines usability as the "extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use." cited in (Henry & Thorp, 2004). In the context of Computer Forensic Analysis Tools we understand usability to be a characteristic of the interaction between the Forensic Investigator and the computer system they are utilising to effect an investigation. What is variable is the task they are attempting to achieve and the environment in which they are working to achieve this goal.

Nielsen sees usability within the whole context of System Acceptability (Nielsen, 1993). Usability is a characteristic that relates to Usefulness. It differs from Utility as that measure relates to can the required function be performed at all. For example, can the deleted files be recovered in part or in whole. Nielsen splits usability into five areas (op. cit.). Learnability relates to how easily a newcomer to a product can use it. It can be measured by the time it takes to learn to use a feature at a stated performance level. Efficiency is the prime measure of effort a competent user requires to complete a task. Its usual measure is the time needed to complete a task. Memorability relates to performance when a user returns to a system after a period of non-use. Error Rate and Severity relates to frequency of deviations from the correct course of actions to complete a task (whether intended or not), with severity measuring the effect that these deviations have on the user and task. Finally, the measure of Subjectively Pleasing recognises that human endeavour should leave the user in a positive frame of mind, at least as far as the use of the tool is concerned. Highly usable systems would be quick to learn, quick to use, slow to forget, usable without error of any kind and result in happy users – as well as completing the task. Of course, this ideal is not always possible and we find that these dimensions of usability are not orthogonal, redesigning a system to improve one might adversely

affect another. For example improving Learnability often reduces Efficiency. The purpose of usability science is to locate issues that negatively affect usability and to try to provide information to designers to change the way products work so as usability is improved.

2.1. Usability of Forensic Tools

It is common for software products to make claims about their usability. For example, Computer Cop Software states that 'All of our products are developed on the philosophy that sophisticated technology can and should reside in an easy-to-use interface' (ComputerCOP Software, 2006). However, there are few actual usability reviews of forensic software that have been published, so it is hard to determine the quality of the justification for such claims. One notable, recent exception is (LaVelle & Konrad, 2007). We wish to continue in that vein.

2.2. Measuring Usability

For our purposes not only is it important to be able to define usability, but also to evaluate usability. For two functionally equivalent tools, the more usable will assist a forensic investigator in completing an examination more. Unless we can evaluate how usable a tool is, we have no basis on which to argue for improved usability. This begs the question, 'how do we evaluate usability'? There are two main broad churches of evaluation. The first uses people representative of users performing a set of typical tasks on the system while measurements are taken. The other tries to predict performance based on evaluation utilising experts in a particular technique. The former methods, normally termed user- or participant-based evaluations are usually thought of as the 'gold standard', in that they are believed to most closely replicate actual behaviour of a system in the field. However, this is not to say that the second system is without benefit. User-based evaluation for simple systems can be expensive and time-consuming. For complex systems, where expertise has to be gained over a considerable length of time, user-based evaluation may be too costly. It is here, and with early design of new systems that expert-based evaluation methods are of benefit. There are debates about the efficacy of expert-based evaluations, but there is evidence that both of these methods find usability issues, even though they might 'find' issues which in fact are not issues for real users of the system or miss others (usability.gov, undated). The usability community accept the recognised limitations of these methods for the benefits they can provide. It is the expert-based evaluation methods that we will be utilizing in this review of Autopsy Forensic Browser.

There are many different expert-based usability review methods, each with their own specific goals within the general goal of finding and reporting usability issues with a product. Two major methods of expert-based evaluation are Cognitive Walkthrough (Wharton et al., 1994) and Heuristic Evaluation (Nielsen, 1993), with Heuristic Evaluation being the most prominent in use.

Heuristic Evaluation is an evaluation method which uses a small set of general rules, such as 'Strive for Consistency' (Shneiderman & Plaisant, 2005), which have to then be interpreted by an evaluator in the light of their experience of using the system. When using the system, do they feel that there are areas which do not support a consistent approach – and do these fail to reach the appropriate levels of striving. This is a difficult task and relies heavily on a broad knowledge of design issues, usability theory and practical experience on the part of the evaluator. Heuristic Evaluation in itself does very little to guide the evaluator in how to go through a process of evaluation.

Cognitive Walkthrough, on the other hand is a more guided process, with a few simple stages to go through. After defining the users of a system in broad terms, the evaluator defines a set of tasks, each with a sequence of actions required to complete them in the system. At each step of this sequence the evaluator asks the questions of himself:

a. Will the users be trying to produce whatever effect the action has?
b. Will users be able to notice that the correct action is available?
c. Once users find the correct action at the interface, will they know that it is the right one for the effect they are trying to produce?
d. After the action is taken, will users understand the feedback they get?

(Abowd, 1995)

Any negative answers to these questions leads to the identification of a usability issue, with an alternative plausible reason statement being made. For example, for question (b) an evaluator might answer 'No, because the button for performing the action is hidden on a different tab in the dialogue.'

3. Method

In order to evaluate the system's usability, three preliminary tasks were performed.

To begin with, a set of two personas were created to represent the two classes of users of the system. The first of these was a serving law enforcement officer, skilled in the use of forensic software, but new to Linux forensic tools. The second was a student new to the whole domain of computer forensic tools, undertaking their first course in the topic on any operating system.

Secondly, a small set of tasks were created that would be representative of tasks required to do a simple forensic investigation on some 'evidence'. Finally, a computer system was created that was capable of running the Autopsy Forensic Browser. The tool itself, nor the underlying Sleuth Kit, were not installed, as this was one of the tasks to be achieved.

The tasks were then performed, 'in slow motion', using the Cognitive Walkthrough method, with the two personas in mind. Where issues were found these were noted appropriately. The reviewers also kept in mind heuristic usability rules and where

these were found to be broken, notes were also kept about these non-conformances. In the end, the task for recovering deleted data was chosen as a task representative of one that might be performed early on in the usage of Autopsy by a student or serving investigator. All results pertain to this task only, even though the tool is capable of a very much greater variety of tasks.

The results were written up and reviewed to try to find clusters of problems.

4. Results

A number of usability issues were found when reviewing the system. Rather than provide the full transcript of the investigation as part of this paper, this can be made available to interested parties by the authors. We have categorised the issues found into the following taxonomy:

i. Usability problems relating to failure to meet web usability guidelines or other Usability Heuristics
ii. Usability problems relating to the use mismatch between users domain language and the system task language
iii. Error Handling and Prevention
iv. The provision of Help and Documentation
v. Task Analysis and System Ethos
vi. Usability problems imposed by the underlying The Sleuth Kit
vii. Usability problems imposed by the use of the Web System Architecture
viii. Usability problems imposed by the Operating System (OS)

There is a difference between the first four areas and the latter four. Whereas the first four it is relatively simple to enhance the system to make it more usable, the last four would require a re-design, or to use a different technological base. This would be more problematic in being able to quickly provide a simple, portable GUI to The Sleuth Kit toolset. It is well understood in Computing that there will be limitations to the development of systems imposed by the choice of platform and architecture and v. to viii. are examples of this.

Of the problems, some will be applicable to only one of the two personas. In particular, the student persona will be more prone to problems relating to the use of language compared to the current investigator, as their domain vocabulary will be much smaller. However, it is recognised that inductees into an area have the task of learning the domain language to be effective.

In the following sections we give examples from each of usability problems found from each of the areas i. to viii. and also give suggestions as to corrective actions that could be taken to enhance the usability in the area.

4.1. Failure to Meet Web Usability Guidelines

There are a large number of 'rules' that have been built up around the design of web pages. Some of these are authoritative in nature, such as those published by the W3C on accessibility (Caldwell et al., 2007), which others are less commanding, such as the 'three-click rule' for web sites (Usability By Design, 2004).

One example from the system is the use of stylised buttons and upper-case text fonts to present the actions to the user. While there is undoubted benefit to users in providing an attractive user interface, in terms of subjective satisfaction, both the colour choices and font used limit readability from even a short distance away and make it difficult for a user with any kind of sight impairment to read them. This can be seen as countering the general web usability guidelines of Nielsen (2005). The buttons could generally be much larger in physical size (more pixels) and have a larger plain area within them to allow for a larger and more readable font to be used with a greater contrast difference to the background. In terms of Gestalt laws of Pragnänz, is it difficult to differentiate figure and ground, the text from the button background (Bruce et al., 2004. Page 130).

There are other example of the use of font, graphic and layout which would be improved by a systematic design phase that looks at these issues.

4.2. Language Mismatch

Autopsy uses several terms that are perhaps outside of the vocabulary of new users of either forensic tools or the Linux operating system. Examples of some of the terms used by Autopsy that these kinds of users may find difficult to understand include: 'case', 'image', 'host', and 'hash database'. There may also be difficulty in understanding the differences between 'disk' and 'partition'. For example, to properly analyse a floppy disk, 'partition' must be selected as the type of image to analyse. 'Meta' and 'symlink' are also used and although the terms are recognised shorthand for metadata and symbolic link respectively, a new user may not know this.

It could be argued that in order to carry out an analysis of a suspect computer system an investigator should be well-versed in the nomenclature of the subject. It is therefore a 'fact of life' that this must be learned before attempting to carry out any such task. Whilst the authors agree that some of the terms are in common usage amongst computer forensics examiners, the distinction between disk and partition in the example outlined above is strange and could be improved upon: perhaps having an option for floppy disk. Better clarification could also be given in other cases. The shortening of metadata to 'meta' may be due to screen real-estate considerations however the authors can see no real reason for the shortening of symbolic link to 'symlink'.

4.3. Error Prevention and Handling

In the software there are several occasions when error prevention is not considered as deeply as it should be. Error Prevention enhances usability by stopping users from getting into situations where errors affect the user. One example of this is the repeated use of text entry fields for entering the details of files available on the system. A user who mis-types in these will introduce invalid data to the system, meaning that errors may occur much later down the line, which might jeopardise a case. In this situation the obvious solution would be to use a 'Browse' button adjacent to the text entry field, as these are provided by HTML forms as an inbuilt component.

When (ultimately, inevitably) errors do occur, these need to be handled in an appropriate manner. Initially it is useful to try to catch the error as soon as possible as this limits the scope for further work to be based on this error. This in turn reduces the cost of the error to the user. The second principle is to enable the user to return to the correct course of action as soon as possible. This requires that the system provide helpful and complete error information and, where possible, to assist with correcting this error. A situation in Autopsy where this is noticeable is that there is a requirement to enter an investigator name. There is a requirement that this has no spaces in it. However, a user who does enter spaces in a name is not prevented from continuing. It is not until later when they cannot select an investigator that they may realise their error. Even then it is not obvious how this has arisen. A user would have to link the earlier action to the later re-action by the system that are several screens apart.

When errors are trapped, there is good advice available on how to write error messages. For example, Shneiderman and Plaisant (2005. Chapter 12) give advice on phraseology and tone of messages for usability.

4.4. The Provision of Help and Documentation

Three examples can be given in this section:

- *Access to help* is not always available on all pages
- Help is not *context sensitive*
- Help is *system* rather than *task-oriented*

The first problem can be seen on the 'Creating Case' page, where the term 'host' is used which might not be readily understood, but there is no immediate access to a button to bring up the help system. This could be solved by providing the help button on all pages. This could be done by making this part of the Cascading Style Sheet (CSS) template for each page. However, this simplistic approach would have consequences for the following element.

When the help system is displayed, there is no initial piece of information displayed in the content frame of the page. This means users have to search through the index to find the section that will provide help to them. To solve this would require that

each page has (a set of) parameters that are passed to the help system, which determine which, of the many help pages, is displayed initially. This should usually be a page specific to the web page that initiated the help to open. There are several difficulties that would need to be overcome here. It would either be up to the web page on which the help was located to specify which page to open at, or the help system to determine which page the request was sent from and act accordingly. The former of these could be achieved through the use of an individually coded help button on each page, but this counters the general solution to problem one. Both the former and latter solution could be implemented using scripting.

The third area relates to theories on how to most effectively provide help and support. When viewing the help system the contents of the index frame are largely organised by the different sections names of the system rather than the tasks the user would wish to perform. For example, a user would have to look in sections: Case Management, File Analysis and possibly Meta Data Analysis to complete the task of recovering a deleted file. Work by Black et al. (1986), suggests that, what he calls, a minimal approach is a more effective way of providing user support. This is not minimal in the sense of being 'a very small user manual', but minimal in the sense of 'minimising the obstrusiveness of the training materials' (op. cit.) in allowing the users to achieve their goals. This is essentially a task-oriented approach, where training materials are oriented toward completing a task rather than learning about the system. There is a tension in this work that things which are side-issues to completing the task are removed where possible. This can lead to a situation where it is argued that 'education' about underlying concepts is left out at the expense of long-term learning (Draper, 1998). Nevertheless, it is highly regarded method of providing assistance to users.

4.5. Task Analysis and System Ethos

One noticeable change between the way in which The Sleuth Kit and Autopsy are used is that Autopsy is a process driven system. By this we mean that Autopsy requires a user to go through a set of tasks before any kind of initial analysis can take place, whereas the The Sleuth Kit does not impose this in quite the same way. This means that users of Autopsy are required to organise their investigation in the way that this tool requires, rather than any locally imposed policy. Although Autopsy has a very light touch in terms of process, there is an additional work load.

4.6. The Underlying Sleuth Kit

As a command-line tool The Sleuth Kit is an extremely flexible and powerful tool with which to examine digital evidence. This is due to the variety of ways in which an investigator can deploy the individual tools and filter the output using various shell commands and scripting constructs. The way in which Autopsy handles requests to The Sleuth Kit however does not have these advantages. The output given by the command-line tools is taken almost verbatim in some cases for the investigator to peruse. Examples include Image Details being obtained using the 'fsstat' command; File Analysis (particularly for showing deleted files) using 'fls';

metadata information using 'istat'; and recovering or exporting a file using 'icat'. Again this is a limitation in the same vein as 4.5 above in that a particular way of investigating a case is imposed upon the examiner. Choices of what parts of the output to display could be given to the user to remedy the situation.

4.7. The HTML/Perl Architecture

The design for Autopsy is perhaps overly complex: C and Perl making up The Sleuth Kit command-line tools; Perl making up the majority of the Autopsy code, with a web server architecture; and a web browser to view and analyse cases. Part of the reason for the web server architecture with web browser access is to enable analysis of file system images from different systems (sleuthkit.org, 2007b). This allows different investigators to work on the same case from different personal computers. The differences in web browser implementation and the complexity involved with running a web server, imposes HCI constraints on the application that could interfere with an investigation. For example, previewing different types of files requires that the viewers for these files are installed as plug-ins for the browser. A plethora of tests could be carried out to check the suitability of different browsers and how they handle different file types. Solutions could include standardising on a browser or writing a custom application to perform a similar function to the browser.

4.8. The Operating System

Autopsy currently works on the following platforms Linux, Mac OS X, OpenBSD, FreeBSD, and Solaris. It is also possible to get the software to work in a Windows environment using Cygwin (Lucas, 2004). Despite Autopsy working in Windows it only does so as the Cygwin library emulates a UNIX environment. A user of Autopsy therefore must have some understanding of UNIX-like operating systems. Students of the authors' classes find it difficult to get used to some of the peccadilloes of this system: the locations and directory structure; the range and abilities of the applications; and of course the command-line management. It is the authors' opinion that UNIX/Linux has many forensically relevant advantages over its Window counterpart. For example, file systems can be mounted read only, with no execution privileges. It is however possible to hide some of the details of the operating system from the user so that they do not get overwhelmed with complexity.

5. Conclusions and Future Work

We have provided a number of examples from a simple review of the Autopsy system where we believe usability could be improved. The categorisation indicates where there are easy enhancements in this area and where there is less scope for dramatic improvement. It is understood that this review is limited in several ways. First within the review itself it looked at only one small task. Secondly, the methodology can be criticised for not being a usability test with users, which leads to greater subjectivity in the results. However, we strongly believe that the results we found indicate that there is a big scope for improvement in Autopsy within the

screens we have reviewed and this would suggest that there might be similar enhancements to be made elsewhere.

It is hoped to follow this work up by taking our own suggestions and implementing changes in the Autopsy system, still using the current architecture. This work will be made available to the public domain under a GNU General Public License (GPL), as is the current case with the current Autopsy installation. We hope to then retest the two systems against each with a user-based task-oriented usability test.

References

Abowd, G., (1995). Performing a cognitive walkthrough. http://www.cc.gatech.edu/computing/classes/cs3302/documents/cog.walk.html (Accessed Dec 5, 2007).

Black, J.B., Carroll, J.M. And Mcguigan, S.M., (1986). What kind of minimal instruction manual is the most effective, *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface* 1986, ACM.

Bruce, V., Green, P.R. And Georgeson, M.A., (2004). *Visual Perception*. First Reprint. Psychology Press, Taylor and Francis.

Caldwell, B., Cooper, M., Guarino Reid, L. and Vanderheiden, G., (2007). 2007, Web Content Accessibility Guidelines 2.0. http://www.w3.org/TR/WCAG20/ (Accessed Dec 5, 2007).

Carrier, B., (2003). Open Source Digital Forensic Tools: The Legal Argument. http://www.digital-evidence.org/papers/opensrc_legal.pdf (Accessed Dec 11, 2007).

Computercop Software, (2006). Computer Cop. http://www.computercop.com/ (Accessed Dec 11, 2007).

Cygwin, 2007. Cygwin. http://www.cygwin.com/.

Draper, S.W., (1998). Practical problems and proposed solutions in designing action-centered documentation. In: J.M. Carroll, ed, *Minimalism beyond the Nurnberg funnel.* MIT Press, pp. Chapter 13.

Farmer, D. and Venema , W., (1999). The Coroner's Toolkit (TCT). http://www.porcupine.org/forensics/tct.html. (Accessed Dec 5, 2007)

Gottschalk, L., Liu, J., Dathan, B., Fitzgerald, S. And Stein, M., (2005). Computer Forensics Programs in Higher Education: A Preliminary Study. *ACM SIGCSE BULLETIN,* **37**(1), pp. 147-151.

Henry, S.L. and Thorp, J., (2004). Notes on User Centered Design Process (UCD). http://www.w3.org/WAI/redesign/ucd (Accessed Dec 5, 2007).

Lavelle, C. And Konrad, A., (2007). FriendlyRoboCopy: A GUI to RoboCopy for computer forensic investigators. *Digital Investigation,* **4**(1), pp. 16-23.

Lucas, C., (2004). Running Sleuthkit and Autopsy Under Windows. http://www.sleuthkit.org/sleuthkit/docs/lucas_cygwin.pdf (Accessed Dec 7, 2007).

Nielsen, J., (2005). Top Ten Web Design Mistakes of 2005. http://www.useit.com/alertbox/designmistakes.html (Accessed Dec 5, 2007).

Nielsen, J., (1993). Usability Engineering. Morgan-Kaufmann.

Shneiderman, B. and Plaisant, C., (2005). Designing the User Interface: Strategies for Effective Human-Computer Interaction. Fourth edn. Addison Wesley.

sleuthkit.org, (2007a). Autopsy Forensic Browser. http://www.sleuthkit.org/autopsy/desc.php (Accessed Dec 13, 2007).

sleuthkit.org, (2007b). The Sleuth Kit. http://www.sleuthkit.org/sleuthkit/desc.php (Accessed Dec 13, 2007).

Usability By Design, (2004). Usability Glossary: 3 Click Rule. http://www.usability.uk.com/glossary/3-click-rule.htm (Accessed Dec 13, 2007).

Usability.gov, (undated). Heuristic Evaluation. http://www.usability.gov/methods/heuristiceval.html (Accessed Dec 7, 2007).

Wharton, C., Rieman, J., Lewis, C. and Polson, P., (1994). The Cognitive Walkthrough: A practitioner's guide. In: J. Nielsen and R.L. Mack, eds, *Usability Inspection Methods*. John Wiley and Sons.