Evaluating Information Visualisations

Keith Andrews
IICM, Graz University of Technology
Inffeldgasse 16c
A-8010 Graz, Austria
kandrews@iicm.edu

ABSTRACT

As more experience is being gained with the evaluation of information visualisation interfaces, weaknesses in current evaluation practice are coming to the fore.

This position paper presents an overview of currently used evaluation methods, followed by a discussion of my experiences and lessons learned from a series of studies comparing hierarchy browsers.

1. INTRODUCTION

As the information visualisation community matures, the evaluation of information visualisation techniques is becoming more prevalent. However, running meaningful evaluations of information visualisation interfaces is an intricate process and recent publications [6, 19] have identified several weaknesses in current evaluation practice.

From my own experience, I believe the following to be the main weaknesses in current information visualisation evaluation practice:

- *Buggy implementation of new interfaces*: New information visualisation interfaces tend to be buggy and incomplete. It seems unfair to then compare them to tried and tested traditional interfaces.

- *Testing the wrong users*: Many studies of information visualisation interfaces seem to test computer science or other students, presumably for reasons of cost and convenience. It might be more beneficial to test real users of information visualistion techniques, such as pharmaceutical or intelligence analysts, but availability considerations often preclude this.

- *Familiarity of test users with traditional interfaces*: Even after extended training of test users with a new information visualisation interface, it is extremely difficult to overcome the bias caused by familiarity with traditional interfaces.

BELIV 2006 Venice, Italy

- *Testing the wrong tasks*: For convenience, many studies of information visualisation interfaces seem to test simple locate, count, and compare tasks. Some of the real benefit from information visualisation systems comes from a deeper insight gained from much more involved exploratory tasks, but these are difficult to test.

2. EVALUATION METHODS

Evaluation methods can be divided into *inspection methods*, where specialist evaluators inspect an interface and use their experience and judgement to assess it, and *testing methods*, where representative end users use one or more interfaces and observations or measurements are made [1].

Often, studies of information visualisation interfaces test computer science or other students, presumably for reasons of cost and convenience. It might be more beneficial to test real users of information visualisation techniques, such as pharmaceutical or intelligence analysts, but considerations of availability and cost often preclude this.

There is some discussion about which kinds of tasks to test. Many infovis studies seem to test simple locate, count, and compare tasks. Some of the real benefit from information visualisation systems comes from a deeper insight gained from much more involved exploratory tasks, but such insight tasks are extremely difficult to formulate and thus test.

The three main kinds of testing methods are formative tests, summative tests, and usage tests.

2.1 Formative Testing (Thinking Aloud Tests)

Formative testing usually involves observing a small number (3–5) of test users using an interface in order to gain insight into which problems occur and *why* they occur [17]. This emphasis on why information (or *process data*) distinguishes formative testing from other kinds of testing.

The classical type of formative test is the *thinking aloud test*, where users are asked to verbalise their thoughts while using the interface. A thinking aloud test is a valuable technique during the development of an information visualisation to gain design feedback and fix bugs.

While thinking aloud testing should be an integral part of system design, the results from one thinking aloud test are rarely generalisable to other information visualisation projects, for two reasons. First, the number of test users is usually far too small to be considered a representative sample and second, the data gathered is process data giving insight into problems found while performing specific tasks

using one particular interface.

Furthermore, evidence shows that users who are thinking aloud both change their behaviour and are slower than users who do not think aloud [7, page 105]. Hence, it makes no sense to measure usability attributes such as effectiveness or task completion time during a thinking aloud test. Such measurements only make sense in the setting of a formal experiment, without any thinking out loud.

2.2 Summative Testing (Formal Experiments)

Summative testing involves the collection of *bottom-line* measurement data, such as task completion time or number of clicks, followed by more or less rigorous statistical analysis [10, 17]. The classical kind of summative test is the *formal experiment*. When comparing two or more information visualisation interfaces, there are two ways to design the formal experiment. A *between-groups* experiment uses independent groups of test users, where each group of test users tests one interface. A *within-groups* (or *repeated measures*) design uses one group of test users, where every uses tests each interface, but the presentation order of the interfaces is counterbalanced. A repeated measures design is often chosen, because the same users are "re-used" for each interface, eliminating any effect of individual differences between users, and making it possible to detect significant differences with fewer test users.

Some researchers have tested their own in-house implementations of information visualisation techniques, leading to criticism that some of the techniques may have been suboptimally implemented. Other researchers have preferred to contact the original developers of a particular technique to obtain original implementations [14]. I believe that there are advantages and disadvantages to both approaches and that they more or less balance each other out. Original implementations are often built to different levels of quality, often in different programming languages, and sometimes requiring external packages to run. Their data formats may be different, requiring substantial work to implement the same test data in each system. At least with in-house implementations the researcher has some control over the quality of each implementation.

Many information visualisation studies use a repeated measures design and report their statistics in terms of analysis of variance (ANOVA) and pairwise t tests. However, according to many statistics textbooks, these tests should only really be performed if the measured values follow a (sufficiently) normal distribution [9, page 49]. In most of the information visualisation studies I have been involved in, the measurement data has not been normally distributed, nor anywhere near normally distributed. Time measurements in particular tend to be skewed to the left, with faster times clumped together and a larger variety of slower timings with outliers. Although some of the statisticians I have consulted regard ANOVA to be robust enough to cope with non-normal data, to be absolutely sure non-parametric statistical tests such as Friedman's test should be used.

When planning a formal experiment, the number of test users (sample size) is key. I have rarely found any statistically significant differences in performance data with less than 10 test users, and usually not with 16–20 or even 32 test users, depending on the sizes of any differences in the interfaces being compared. It would seem necessary to plan for a minimum of 50 or possibly even 100 test users, to have a fair chance of discovering differences with statistical significance in performance data. I have, however, found statistically significant differences in subjective ratings with 16–20 test users or less.

2.3 Usage Studies

Usage studies involve observation and/or recording of users over a longer period of time working with an interface [13]. A recent workshop at CHI 2005 [15, 16] discussed usage studies in depth.

Many usage studies seem to rely on self-reporting, i.e. users keeping a diary or log of what they did, what happened, and how long they spent doing various things [5, 21]. I am very sceptical as to the accuracy of self-reported data, and much prefer the alternative of recording and manually transcribing user events, even if manual coding from video and log files is *extremely* time-consuming [3, 4].

In Graz we have just finished a usage study of four users of office software. Each user was recorded for several hours on a typical working day, and their activities were then transcribed manually using special coding software and codes from a carefully designed coding taxonomy. For each hour of video, approximately ten hours were spent manually coding user events. The analysis revealed some very interesting and surprising patterns of use. For example, the university secretarial staff in our study spent 27.5% of their computer time in the university information system, 24.5% in spreadsheets, 22.7% in word processing, and 14.2% in email. They also spent 53.6% of their waiting time waiting for SAP [18].

Usage studies are well-suited to learning how users use a piece of software, but are not useful for objective comparison of two or more interfaces. There are also very few reported usage studies of information visualisation software [12].

3. COMPARING HIERARCHY BROWSERS (THE TREE VIEW ALWAYS WINS)

In 2002 we performed a study comparing our new-fangled InfoSky galaxy view browser to a traditional windows explorer style tree view browser [2] for documents in a hierarchy. The galaxy view browser used recursive voronoi tesselations to subdivide polygonal regions representing the hierarchy containing the documents, as can be seen in Figure 1.

Initial feedback from early users was very positive, but users still seemed to prefer using the tree view browser over the galaxy view. Anecdotal evidence suggested this was because of prior familiarity with the tree view. When we compared the galaxy view directly one-to-one against the traditional tree view in a formal experiment, it did indeed perform rather badly, being significantly slower on every task we measured.

Numerous bugs and problems in the galaxy view were fixed and a second study was run [11] on the revised interface shown in Figure 2. In the second study, the tree view was still faster than the galaxy view for every task, although not all of the differences were statistically significant. Thus, the galaxy view "advanced" from being significantly slower than the tree view to there being no significant difference on most tasks, although I hesitate somewhat to claim this as progress.

In our most recent study, four hierarchy browsers were compared in a counterbalanced repeated measures study

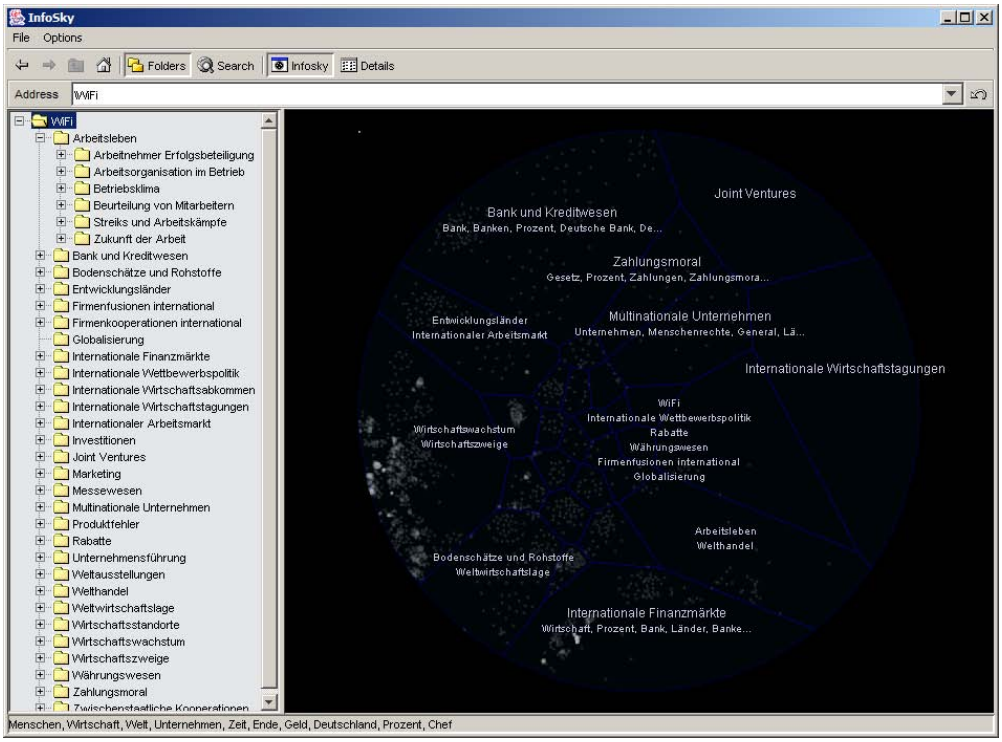

Figure 1: The InfoSky system. On the left, the InfoSky tree view browser, similar in look and feel to a traditional windows explorer browser. On the right, the InfoSky galaxy view.

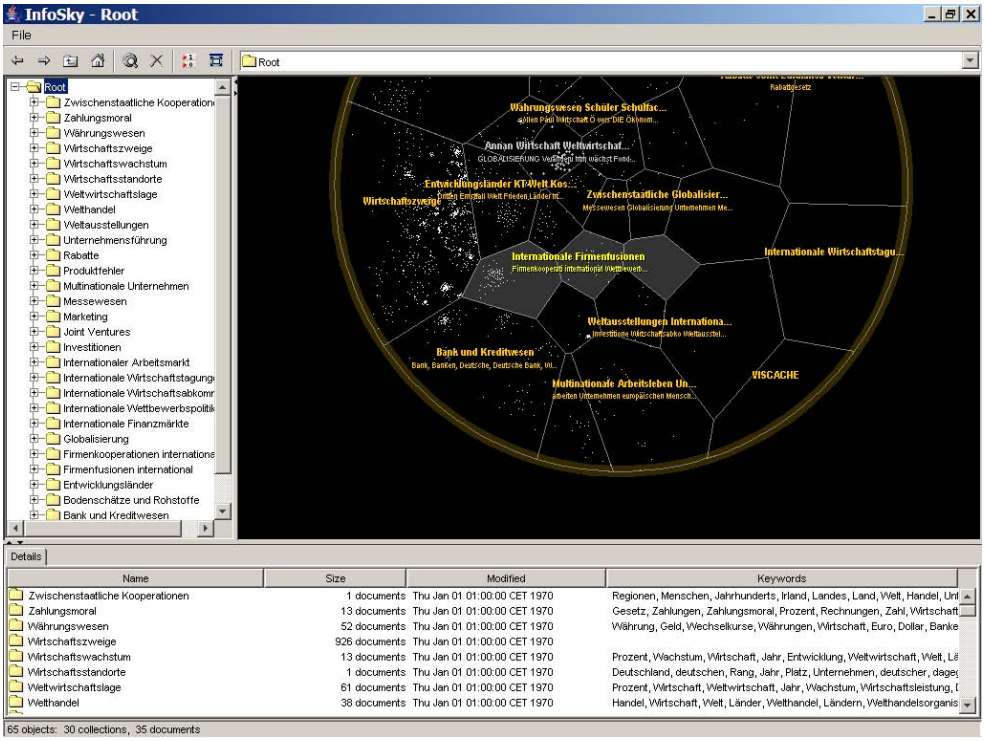

Figure 2: InfoSky 2.0. After user feedback, among many other improvements, the thin blue polygon edges in the galaxy view were made orange.

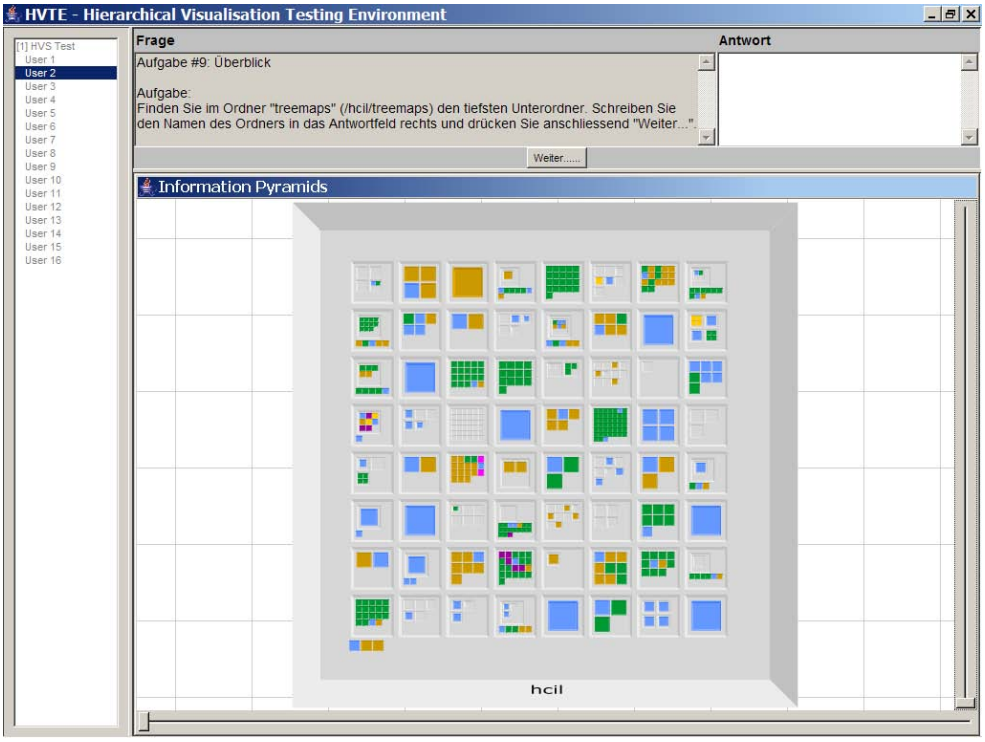

Figure 3: The Hierarchical Visualisation Testing Environment (HVTE), an semi-automated environment for testing hierarchy browsers.

with 32 test users. The four browsers tested were in-house implementations of 1) a windows explorer style tree view, 2) an information pyramids browser, 3) a tree map browser, and 4) a hyperbolic browser. The browsers are part of the Hierarchical Visualisation System (HVS) [20]. The Hierarchical Visualisation Testing Environment (HVTE) is a semi-automated test environment built on top of HVS (see Figure 3). HVTE presents tasks, browsers, and a test hierarchy for a particular test case to the user and automatically gathers timing data.

The test hierarchy used was the reduced version of the logs A hierarchy from the InfoVis 2003 contest [8]. The original logs A hierarchy contains about 70,000 leaf nodes. The reduced hierarchy is the hcil subtree (everything under /projects/hcil) containing 3,239 leaf nodes, which represents part of the file system of the University of Maryland Computer Science Department web site.

Each user performed eight tasks with each browser. The eight tasks were divided into two overview tasks, two search tasks, two count tasks, and two compare tasks. Four sets of equivalent tasks were designed (8 tasks in each set). The tasks were formulated in the native language of the test users (German).

The task completion data revealed no statistically significant differences between the four browsers, except in one case: the tree map browser was significantly faster than the hyperbolic browser for the task involving the counting of files in a subdirectory. In the subjective ratings, test users consistently rated the tree view browser significantly higher than the other browsers in a variety of factors. The tree map browser was consistently rated significantly lower than the other browsers in a variety of factors.

4. CONCLUDING REMARKS

Looking at the InfoSky and the hierarchy browser studies, it seems at first glance that we might as well give up. Whatever new-fangled visualisation we build, when we test it against the tree view, the tree view always wins. Even if our performance data show no significant differences, users significantly prefer the tree view. Users will apparently need a great deal of persuading to move from a familiar trusted interface to a new, unfamiliar one.

Formative techniques such as thinking aloud testing are well-suited to providing development feedback when building an information visualisation system. Usage studies can be used to gather usage data once a system has been built and is in use. However, neither formative testing nor usage testing are suitable for objective comparison of two or more infovis techniques. For comparative studies, formal experiments offer the only objective solution.

Perhaps a few steps in a strategy to move forward would include:

- Looking for experienced analysts to act as test users, rather than computer science students.
- Providing extensive training in each of the infovis techniques under trial, to counter any bias arising from prior experience.
- Formulating more involved tasks which more accurately reflect the kind of exploration and analysis tasks for which infovis systems excel.

Automating the running of formal experiments as far as possible (as in the HVTE system) is also a step toward making comparative evaluations more feasible.

5. ACKNOWLEDGMENTS

Michael Granitzer, Wolfgang Kienreich, and Vedran Sabol and several other colleagues from the Know-Center Graz and Hyperwave participated in the development and evaluation of InfoSky. Werner Putz, Alexander Nussbaumer, and Johann Stampler built the Hierarchical Visualisation System (HVS), which Janka Kasanicka extended to build the Hierarchical Visualisation Testing Environment (HVTE), used for the comparative study of four hierarchy browsers.

6. REFERENCES

[1] K. Andrews. *Human-Computer Interaction: Lecture Notes*. 2006. `http://courses.iicm.edu/hci/hci.pdf`.

[2] K. Andrews, W. Kienreich, V. Sabol, J. Becker, G. Droschl, F. Kappe, M. Granitzer, P. Auer, and K. Tochtermann. *The InfoSky Visual Explorer: Exploiting Hierarchical Structure and Document Similarities*. *Information Visualization*, 1(3/4):166–181, December 2002. doi:10.1057/palgrave.ivs.9500023.

[3] M. D. Byrne, B. E. John, and E. Joyce. *A Day in the Life of Ten WWW Users*, 2000. `http://citeseer.ist.psu.edu/400156.html`. Unpublished paper.

[4] M. D. Byrne, B. E. John, N. S. Wehrle, and D. C. Crow. *The Tangled Web We Wove: A Taskonomy of WWW Use*. In *Proc. Conference on Human Factors in Computing Systems (CHI'99)*, pages 544–551. ACM, Pittsburgh, Pennsylvania, USA, May 1999. ISBN 0201485591. doi:10.1145/302979.303154.

[5] I. Ceaparu, J. Lazar, K. Bessiere, J. Robinson, and B. Shneiderman. *Determining Causes and Severity of End-User Frustration*. Technical Report CS-TR-4371, University of Maryland, Computer Science Department, May 2003. `http://hcil.cs.umd.edu/trs/2002-11/2002-11.pdf`. Revised Version.

[6] C. Chen and Y. Yu. *Empirical Studies of Information Visualization: A Meta-Analysis*. *International Journal of Human-Computer Studies*, 53(5):851–866, November 2000. doi:10.1109/6.469330.

[7] K. A. Ericsson and H. A. Simon. *Protocol Analysis: Verbal Reports As Data*. MIT Press, revised edition, May 1993. ISBN 0262550237.

[8] J.-D. Fekete and C. Plaisant. *Information Visualization Benchmarks Repository*. 2003. `http://www.cs.umd.edu/hcil/InfovisRepository/contest-2003/`.

[9] A. Field. *Discovering Statistics Using SPSS for Windows*. Sage Publications, 2000. ISBN 0761957553.

[10] A. Field and G. Hole. *How to Design and Report Experiments*. Sage Publications, 2002. ISBN 0761973834.

[11] M. Granitzer, W. Kienreich, V. Sabol, K. Andrews, and W. Klieber. *Evaluating a System for Interactive Exploration of Large, Hierarchically Structured Document Repositories*. In *Proc. IEEE Symposium on Information Visualization (InfoVis 2004)*, pages 127–134. Austin, Texas, October 2004. doi:10.1109/INFOVIS.2004.19.

[12] E. Hetzler and A. Turner. *Analysis Experiences Using Information Visualization*. *IEEE Computer Graphics and Applications*, 24(5):22–26, September/October 2004. doi:10.1109/MCG.2004.22.

[13] D. M. Hilbert and D. F. Redmiles. *Extracting Usability Information from User Interface Events*. *ACM Computing Surveys*, 32(4):384–421, December 2000. doi:10.1145/371578.371593.

[14] A. Kobsa. *User Experiments with Tree Visualization Systems*. In *Proc. IEEE Symposium on Information Visualization (InfoVis 2004)*, pages 9–16. Austin, Texas, USA, October 2004. doi:10.1109/INFVIS.2004.70.

[15] J. Kort. *CHI2005 Workshop 6 Results*. April 2005. `http://usage.nl/workshop.html`.

[16] J. Kort and H. de Poot. *Usage Analysis: Combining Logging and Qualitative Methods*. In *Proc. CHI2005 Workshop*. ACM, Portland, Oregon, USA, April 2005. doi:10.1145/1056808.1057117. `http://www.usage.nl/workshop.html`.

[17] T. Landauer. *Research Methods in HCI*. In M. G. Helander, editor, *Handbook of Human-Computer Interaction*, chapter 42, pages 905–928. North-Holland, 1988. ISBN 0444705368.

[18] G. Modes. *A Usage Study of Desktop Users: Technical Realisation and Coding Software*. Master's thesis, Graz University of Technology, Austria, September 2005.

[19] C. Plaisant. *The Challenge of Information Visualization Evaluation*. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI 2004)*, pages 109–116. May 2004. doi:10.1145/989863.989880.

[20] W. Putz. *The Hierarchical Visualization System: A General Framework for Visualizing Information Hierarchies Using the Example of Information Pyramids*. Master's thesis, Graz University of Technology, Austria, March 2005. `ftp://ftp.iicm.edu/pub/theses/wputz.pdf`.

[21] A. J. Sellen, R. Murphy, and K. L. Shaw. *How Knowledge Workers Use the Web*. In *Proc. Conference on Human Factors in Computing Systems (CHI 2002)*, pages 227–234. ACM, Minneapolis, Minnesota, USA, April 2002. doi:10.1145/503376.503418.