Data Transfer: A Longitudinal Analysis of Clipboard and Drag-and-Drop Use in Desktop Applications

Jonathan Woodruff, Jason Alexander*

School of Computing and Communications, InfoLab21, South Drive, Lancaster University, Lancaster LA1 4AX, United Kingdom

Abstract

Data transfer within and between desktop applications facilitates efficient data-centric tasks on modern computer operating systems. This data can be transferred via the clipboard (cut, copy, paste) or through more direct drag-and-drop actions. This paper presents results gathered during a 90-day longitudinal log study of 17 participants' clipboard and drag-and-drop interactions. The paper characterises the frequency, time and type of actions, interaction mechanism, and whether the users' computer configuration affects these findings. We found clipboard operations are more common than drag-and-drop operations (and drag-and-drop is rarely used by some participants), most data transfer occurs on the same screen for multi-screen users, clipboard and drag-and-drop operations are used for different applications and the applications used for clipboard operations align with previously reported results.

Keywords: Clipboard, Copy-and-Paste, Drag-and-Drop, Desktop Interaction, Data Transfer, Multiple Screens

1. Introduction

Cut, copy, and paste are the universally recognised commands to transfer data within and between desktop applications; drag-and-drop provides a direct-manipulation equivalent. These commands significantly increase user efficiency and accuracy by allowing quick replication or transfer of data or objects. Typically, this data takes the form of text, images, or complete files. Despite their widespread adoption, there is no large-scale empirical characterisation of how these tools are utilised in everyday computing.

This knowledge gap means human-computer interaction researchers lack empirical evidence to direct and drive development of new data transfer tools and techniques. Specifically, we lack an understanding of these tools' regularity and patterns of use, their use in different applications, and the types of data transferred. This work seeks to fill this knowledge gap. We see two core areas benefiting from this new understanding: (1) the improvement and development of novel data-transfer tools and; (2) the development of cross-application, task-centric interaction tools—data transfer creates an implicit link between source and destination applications. Improvements in these areas could have significant impact: data transfer is a ubiquitous action in all forms of desktop computing.

To improve our understanding of data transfer actions, this paper aims to characterise clipboard and drag-and-drop use with unmodified applications in a desktop computer environment. Previous work in this space focused on shorter periods of time [1], specific user groups [2], or developing new interaction techniques [3, 4]. Our work goes beyond that in the literature by evaluating data transfer over a 3 month (90 day) period, includes the first characterisation of drag-and-drop behaviour, and assesses the changes in user behaviour since the last report of 'in the wild' data transfer from 2009 [1].

To achieve this goal, we used the MultiLog [5] software for data collection and undertook a longitudinal log study for a period of 90 days with 17 participants. This period allowed us to collect long-term behavioural data and everyday interaction with many applications: something not possible in short-term observations. Overall, we collected 31,567 cut, copy and paste actions and 3,432 drag-and-drop actions across all of our participants. Our analysis of this data found that (1) Clipboard operations are used more frequently than drag-and-drop; (2) Most data transfers occur across only one screen, the choice of drag-and-drop vs. clipboard is application dependent; (3) Text was the most frequently transferred item on the clipboard; (4) Keyboard initiation is more frequent than mouse initiation for clipboard operations.

To summarise, this paper contributes: (1) A 90-day, 17 participant log study of data-transfer within and between desktop applications—the longest study of this type; (2) An empirical characterization of data-transfer in desktop applications (copy-and-paste, drag-and-drop), including the first published insights into how drag-and-drop is used 'in the wild'; (3) An analysis and discussion of the implications of our findings, including changes in behaviours since previous studies. The paper proceeds to describe relevant related work, the longitudinal study methodology, the characterization, and finishes with a discussion of the implications of this work.

*Corresponding author.
Email addresses: j.woodruff@lancaster.ac.uk (Jonathan Woodruff), j.alexander@lancaster.ac.uk (Jason Alexander)

2. Related Work

Relevant related work falls in three key areas: (1) Clipboard (cut, copy, and paste); (2) Drag-and-drop and; (3) Logging software for longitudinal research studies.

2.1. Clipboard (Cut, Copy, Paste)

A cut or copy operation occurs when a user requests (through mouse or keyboard input) text, image(s) or file(s) to be cut (removed from source) or copied (duplicated from the source) to the clipboard (a temporary storage location). A paste operation occurs when the user requests the text, image(s) or file(s) be retrieved from the clipboard and be copied to the destination. Objects remain on the clipboard until over-written by a new copy/cut action.

Stolee et al. [1] detail early work in understanding user interactions with the clipboard. They studied 15 participants' clipboard use over four weeks, reporting the type of applications used and a breakdown of their use as either a source or destination for data. The authors identified two distinct sets of usage patterns: elementary and complex. Elementary patterns were: (1) copying and pasting within the same application or (2) copying and pasting from one application to another. Complex patterns characterised copying from one source to multiple destinations or vice versa. These patterns showed that within application clipboard operations occurred 70% of the time, while between application clipboard operations occurred 30% of the time. Various procedures for automating clipboard operations were suggested, including a multiple-item clipboard, a context-aware clipboard and/or a clipboard with iteration.

Kim et al. [2] explores through a field study how five programmers utilise the clipboard in their work environment over a period of approximately 50 hours. They note that programmers use the clipboard frequently both from within code and document editors. The clipboard can however cause code duplication which is easy to create with the clipboard and difficult to eliminate once present.

In both cases, this work goes beyond these two studies by logging clipboard actions for a longer period (90 days), including drag-and-drop as a data transfer technique, understanding the role of multiple screens, and updating the most recent findings from 2009 [1].

2.2. Drag-and-Drop

A drag-and-drop operation occurs when a user presses the left mouse button on a draggable object[1], moves the mouse to the destination location, and releases the mouse button to "drop" the object into the destination window/file. Drag-and-drop is a frequently used feature of modern operating systems and there has recently been an increase in web-based drag-and-drop to simplify web services and aid users when interacting with websites [6].

Research into drag-and-drop has primarily focused on approaches for enhancing the drag-and-drop experience for end users. Kobayashi and Igarashi [7] present a system that allows users to suspend and resume drag-and-drop operations based on a throw-and-catch metaphor. Drag-and-Pop supported long-distance drag-and-drop by relocating potential drop locations closer to the dragged object [8]. Brewster [9] explored whether the addition of non-speech sounds could increase the usability of drag-and-drop while Shih et al. [10] provided an automatic assistive program to improve the efficiency of drag-and-drop for users with developmental disabilities. By focusing on repetitive or complex tasks, user efficiency can also be increased by simplifying the interaction to include drag-and-drop actions (e.g. to aid users with image composition and photograph labelling [11]).

We found no current reports of how drag-and-drop is used 'in the wild'; this work will contribute that knowledge.

2.3. Logging Software

Logging software allows researchers to automatically collect an empirical account of events that occur on participants' computers. Loggers are generally considered low-level or high-level. RUI [12] is a low-level keystroke and mouse action logger for Windows whereas VibeLog [13], Pylogger [14] and Microsoft PSR [15] are higher-level loggers which collect more contextual information surrounding user actions. Software logging allows large-scale longitudinal data collection, but does suffer from a lack of contextual information and cannot report user intentions [16, 17].

To log clipboard and drag-and-drop data interactions, we employed MultiLog [5], a system that supports "plug in" logging software and the generation of a universal log output. We wrote two new plug-ins—one for the clipboard and one for drag-and-drop (described in the next section).

3. Longitudinal Study Methodology

The overarching goal of this study is to understand how desktop computer users interact with the clipboard and perform drag-and-drop actions during their everyday computer use. To do this, we conducted a longitudinal logging study that recorded user behaviour when using unmodified versions of their applications, on their own computer—the Windows OS logging software captured appropriate events in the background. We used the MultiLog [5] software which allows existing logging applications to be "plugged in" to gather the required data set.

3.1. Participants

Seventeen computer users from a variety of different career backgrounds took part in the longitudinal log study for a period of 90 days. Their occupations were: PhD student (3 Computer Science (CS), 2 Design), CS lecturers (3), CS undergraduate students (3), and one each of Freelance blogger, Software Engineer, researcher, personal assistant, IT systems manager, and retired. 11 of the participants were frequent computer users and 5 of the overall participant pool were female. The age range of participants was between 18 and 74 years. Seven users had single screens, while eight had two, and two users had three

[1] Including 'user defined' draggable objects such as highlighted text.

screens[2]. Eight participants used Windows 7, three used Windows 8.1, and three used Windows 10. Seven users had laptops, while eleven had desktops. Three participants used their computer for home/leisure tasks, five for work-only tasks, and the remainder used it for both. All participants started the study within the first 16 days of September, 2015 and all continued for a 90 day period.

3.2. Apparatus

The MultiLog software [5] starts on user login, automatically logs all copy-and-paste and drag-and-drop operations, and intermittently uploads these to a server when an Internet connection is present. The user can pause logging at any point through an icon in the system tray (the act of pausing is not logged to ensure user privacy).

The MultiLog architecture supports custom "plug-in" loggers. For this study, we developed clipboard and drag-and-drop loggers using the Windows UI Automation API [18].

3.2.1. Clipboard Logging

The clipboard logger records all cut, copy, and paste operations and: a timestamp, the type and number/length[3] of the item(s) copied/pasted, the method used to copy/paste the item, the name, process name, and ID of the source and destination windows, and various spatial information such as the height, width as well as left, right, top and bottom co-ordinates of the source and destination windows.

Clipboard actions are logged through a combination of the `MouseHookListener` and `KeyboardHookListener` classes. These classes allow the global monitoring of mouse and keyboard actions, regardless of application, which are then used to detect clipboard operations the user has performed. For events detected on actionable objects (buttons, menu items) by the `MouseHookListener`, the associated Windows UI Automation object is text queried to determine whether it is a *cut*, *copy* or *paste* operation. This method of querying ensures we capture all forms of these operations, for example, Microsoft Word's "Paste Special ..." action, but does not allow us to separate them in later analysis.

The `KeyboardHookListener` class is used to monitor for the key combinations $Ctrl + X$ (cut), $Ctrl + C$ (copy) and $Ctrl + V$ (paste). Once detected, if a cut or copy operation has occurred, the contents of the Windows Clipboard is queried to gather information as to whether text or files were involved in the action. If text has been cut/copied, the number of characters is recorded and if files have been cut/copied, the number of files is recorded. If text/files have been pasted through a detected paste operation, the text length or the number of files is also recorded in the same way. An example of a clipboard log line is shown in Table 1, row 1.

[2]We did not detect any change in the number of screens through the study period.

[3]For text, this is the number of characters; for files, this is the number of files; information about other types of objects is typically unavailable.

3.2.2. Drag-and-Drop Logging

The drag-and-drop logger records all drag-and-drop actions (these could be file drag-and-drops or the user dragging text or other objects to different locations on screen), it includes: a timestamp, the name of the item dragged, the source window (including process name and ID), the object name it was dragged onto, the destination window (including process name and ID) and the length of time the action took from depressing the mouse button to releasing it.

Drag-and-drop actions also used the `MouseHookListener` class to detect the mouse button being pressed. The item the mouse was pressed on is recorded through the Windows UI Automation framework. The logger then detects mouse movement across the screen while the mouse button is still depressed and records the item (through Windows UI Automation) the mouse button was released on. The source and destination windows are also recorded through the `GetForegroundWindow()` method along with the time the drag and drop operation took. An example of a clipboard log line is shown in Table 1, row 2.

3.2.3. Logger Testing

We conducted significant in-house and beta testing of our plug-ins before we commenced the full study. First, this involved functional testing by the authors of pre-defined use-cases (combinations of cut, copy, paste, and drag-and-drop actions) both within and between a wide range of applications. Test cases were executed and the output examined to ensure matching log lines appeared as expected. This was repeated across the range of Windows Operating Systems that we supported (Windows 7, 8.1, and 10). Second, beta-testing by three colleagues: this ensured that our MultiLog plugins were stable over long periods of time, did not cause noticeable degradation of computing performance, and operated across a wider range of Operating Systems, applications, monitor setups, and contexts of use. This also allowed us to ensure we correctly received the automatically uploaded log files. During this testing phrase we identified two issues with the APIs: (1) occasional repeated logging of single action (e.g. a copy action with all parameters identical, including timestamp, would be recorded twice) and; (2) occasional corrupt log-lines, where incomplete data on an action was reported by the APIs. Both of these issues were beyond our control. We removed these lines in post-processing.

The plug-ins we developed for MultiLog use a third-party API (Windows UI Automation) to collect their respective data. The accuracy of our results is therefore dependent on these APIs. During analysis of the longitudinal study data we removed occasional actions that were erroneous or corrupt.

4. Data Analysis

Following the 90-day study period, we collated the logs and prepared the data set for analysis. The data presented within this paper is a subset of a larger data set which included window switches and other user interaction (mouse, keyboard shortcut) logging. The terms 'window switch' and 'interaction hour' are

1	Item: "TEXT" of size: "591" (characters) COPIED to clipboard (via CTRL + C) from: "Sent - Lancaster - Microsoft Outlook (OUTLOOK (9436))". Window Dimensions: Width: 1200 Height: 961 Position: Left: 1804 Right: 3004 Top: 498 Bottom: 1459
2	Item: "Lecture Information" Dragged from: "Inbox - Microsoft Outlook (OUTLOOK (5604))" to: "Teaching" window: "Inbox - Microsoft Outlook (OUTLOOK (5604))". The action took: 00:00:12.7817703

Table 1: Example log lines. Row 1: Copying text to the clipboard; Row 2: dragging an object within the same application.

used in the results section of this paper. A window switch is an event that is triggered by a switch in focus (by any means) from one active window to another of a different name and/or Operating System process[4]. For timing purposes, we defined an 'interaction hour' as any hour in which any user interaction on the user's computer was detected.

The analysis had multiple stages. First, we filtered superfluous lines from the raw log files, consisting of the duplicate and corrupted lines identified in Section 3.2.2. Second, our post-study analysis of drag-and-drop operations indicated that our logger, as well as collecting legitimate drag-and-drop operations, also collected many operations that *looked* like drag-and-drops but are not. Common examples included attempts to drag 'OK' or 'Cancel' buttons in dialog boxes (even moving the mouse by 1px while the mouse button is pressed results in our logger capturing this behaviour) and highlighting text in a text editor. We built a library of such items for removal and systematically checked all drag-and-drop operations for validity before continuing. We also removed drag-and-drop operations whose duration exceeded one minute (a total of 320 actions). Third, we merged individual log files into one "master" log file per user. Finally, we conducted scripted processing of individual users' data sets. This allowed us to selectively ask questions of the dataset (e.g. "how often do people use the 'copy' command?") and to produce the results detailed in the following section.

5. Results

Our results reporting is structured as follows: (1) Frequency of data transfer; (2) The location and application the data was transferred from/to; (3) Transfer time, type and size and (4) Methods of initiation.

5.1. Frequency of Data Transfer

5.1.1. Clipboard Operations

We observed 31,567 clipboard operations (copy, cut and paste) over the study period, breaking down into 12,677 copy actions (40.1%), 2,680 cut actions (8.4%), and 16,210 paste actions (51.3%). Figure 1 shows the breakdown of these actions, by activation method (keyboard, menu/buttons) for each participant. The total number of actions is shown in the right-hand side, with a diverse spread of use across our participant pool. For the majority of users (14/17), keyboard activation was more frequent that menu activation, although two participants exclusively used the menu for interaction (P01, P10). Overall, these is an almost equal split in data transfer to and from the clipboard.

On average, participants used the clipboard 5.3 times per hour. However, their use was diverse (s.d. 4.2, Figure 2a): eight participants used the clipboard infrequently (on average less than 5 operations per interaction hour), five participants used the clipboard moderately (on average 5–10 times per interaction hour) and four participants utilised clipboard operations on average over 10 times an hour.

5.1.2. Unused Cuts and Copies

An unused cut or copy is a cut or copy operation which is executed by the user but there is no corresponding paste operation. For example, if a user copies, cuts and then pastes some text, the copy operation is unused, as the data is over-written by the data from the cut operation. This may occur because the user initiates an incorrect action (cut/copy), uses cut as a mode of deletion, or the user forgets about their copy/cut action. We observed an average across participants of 1.6 unused operations per interaction hour (s.d. 1.3, see Figure 2b).

5.1.3. Drag-and-Drop Operations

We recorded a total of 3,432 drag-and-drop operations over the study period. However, unlike clipboard operations where users demonstrated diverse use of this interaction (see Figure 2c, yellow dots represent individual participants), we observed three main groups of use: one participant did not use drag-and-drop, many participants used it infrequently—less than 1 operation per interaction hour (14 participants)—and two users used drag-and-drop over 3 times per interaction hour.

5.1.4. Summary

Overall, we found that, on average, the clipboard is utilised more frequently (5.3 operations per interaction hour) than drag-and-drop (0.6 operations per interaction hour), even withstanding the two operations required to complete a full clipboard interaction. As expected, we saw 'paste' was used more often than 'cut' or 'copy'. The use of drag-and-drop was more variable than the clipboard with some participants (P6) not using it at all and others (P12) using it frequently.

5.2. On-Screen Location & Source/Destination Applications

For each clipboard and drag-and-drop operation, we recorded the on-screen location and the applications in which the operation occurred. We paired copy/cut and associated paste operations together for this section.

[4]Some applications start new process for each window, others open multiple windows from the same process.

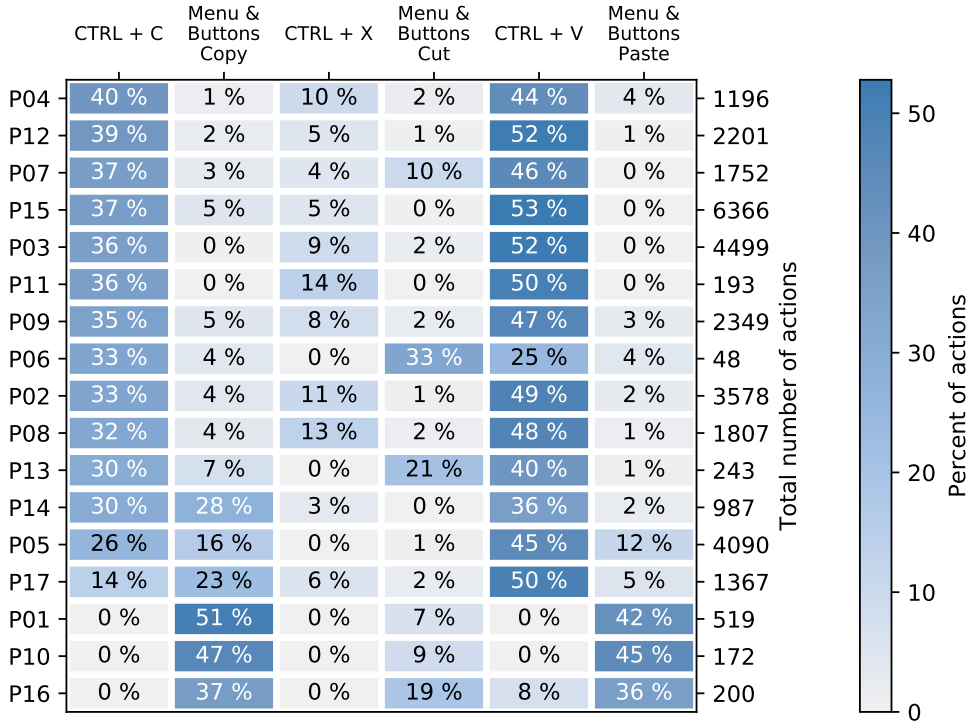

Figure 1: Breakdown of participant's use of clipboard operations, ordered by total use of CTRL-C, percentages rounded to nearest whole value.

5.2.1. Screen Location: Clipboard

For multi-screen users, we found that 81% of all copy/cut and paste pairs occurred on the same screen, with 15% of actions occurring across different screens. 4% of operations were to or from windows that spanned multiple screens. Each group showed high variance between participants (21%, 14%, and 8% respectively); this is show in more detail in Figure 2d.

5.2.2. Screen Location: Drag-and-Drop

For multi-screen users that used drag-and-drop, we found that 98.8% (s.d. 1.8%) of all drag-and-drop operations occurred on the same screen, and only 0.9% (s.d. 1.7%) of actions had a 'drop' location on a different screen to the initial drag. A further 0.3% (s.d. 0.7%) of actions were on windows that spanned multiple screens.

5.2.3. Source/Destination Applications

For each data transfer operation, we logged the source and destination windows and applications. For clipboard operations, 69.5% occurred within the same application (s.d. 3.8%) and for Drag-and-Drop operations, 97% (s.d. 9.1%) occurred within the same application.

To understand in which types of applications this data-transfer occurred, we grouped applications based on their primary function (Figure 3). Drag-and-drop's use was dominant over the clipboard in Presentation Editors (3.5actions/hour, likely due to users positioning elements on screen), spreadsheets (1.9actions/hour, moving data between cells), and in email clients (1.7actions/hour, sorting messages into folders). Programming IDEs showed the most regular use of the clipboard (2.8actions/hour within and 0.8actions/hour between this and other applications). Note that in some cases, actions are only possible (or sensible) with either the clipboard or drag-and-drop (e.g. arranging objects in a presentation).

5.2.4. Number of Window Switches

As part of the study, we logged the number of window switches that occurred between the copy/cut and corresponding paste operation when data was transferred between different applications (Figure 2e). The equivalent data for drag-and-drop does not exist, as these are linear operations and only one window switch can occur (if the participant drops the object in a different window to its source). The average number of switches between the copy/cut and associated paste action was 1.6 (only including instances where at least one switch occurred). The majority of between-window cut/copy and paste actions required only a single window switch (74.4% (s.d. 14.1%)) between the source and destination.

5.2.5. Summary

For both clipboard and drag-and-drop operations, the majority of data-transfer actions occur with the source and destination window on the same screen. When data transfer does occur across multiple screens, it is usually undertaken using the clipboard and in 74% of cases users move directly to the target window. Clipboard operations most regularly occurred in programming IDEs; drag-and-drop operations most regularly occurred in Presentation Editors.

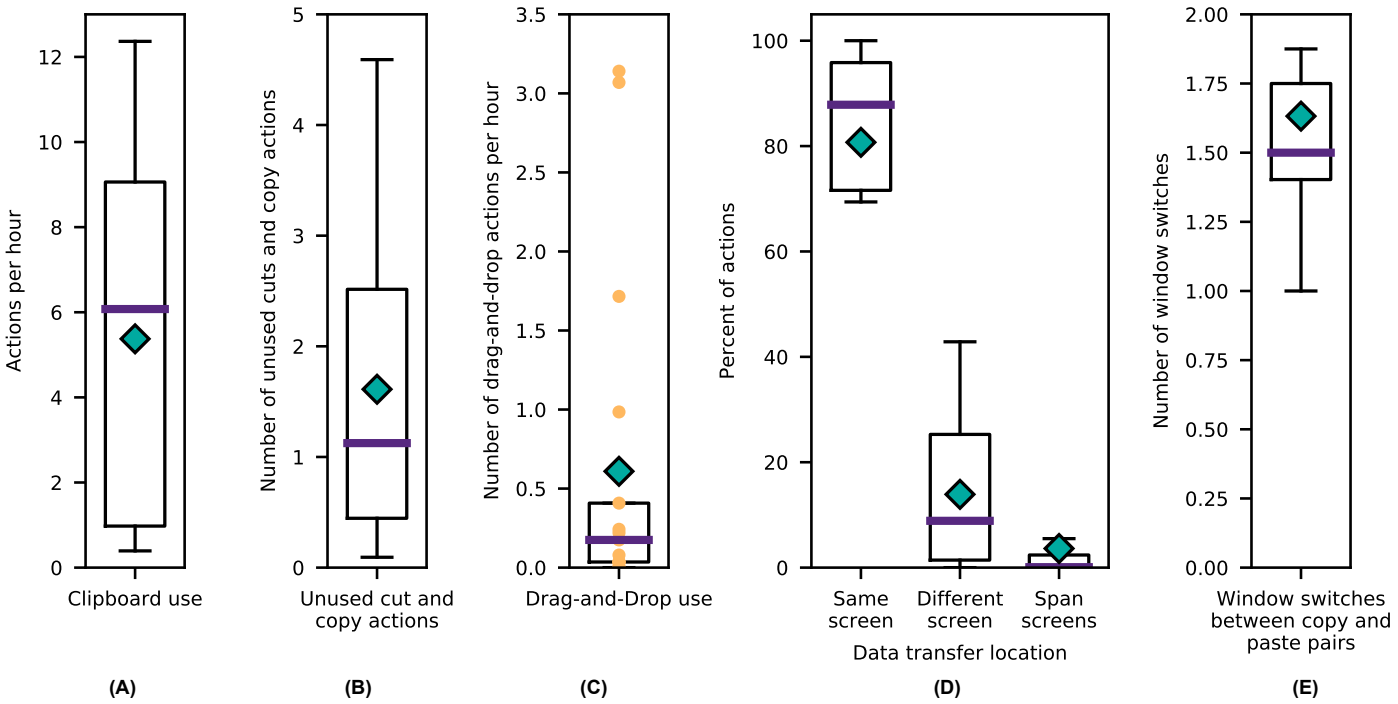

Figure 2: Distribution analyses. Green diamond shows the mean; yellow dots show underlying dataset. (A) Regularity of clipboard operations; (B) Unused cut and copy actions per interaction hour; (C) Regularity of drag-and-drop use; (D) Location of data transfer operations; (E) Number of window switches between cut/copy and past pairs.

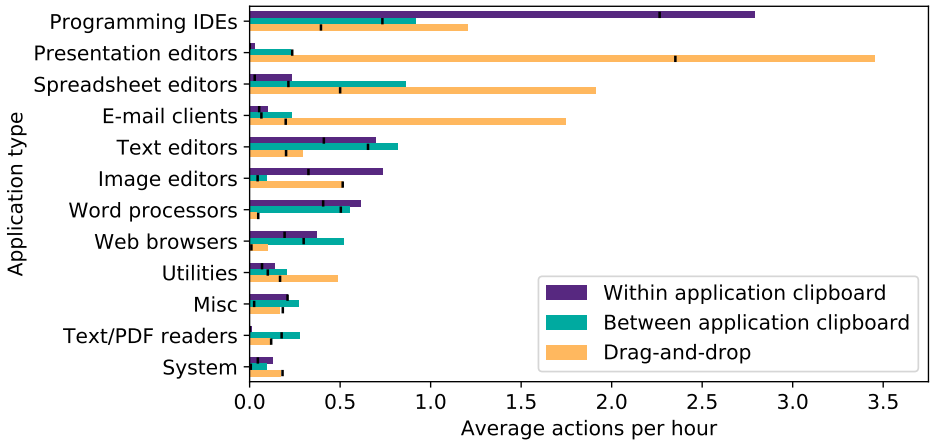

Figure 3: Average clipboard and drag-and-drop operations per hour (black vertical marks indicate median values). Utilities include Windows Explorer, Calculator, and Remote Desktop.

5.3. Transfer Time, Type & Size

5.3.1. Transfer Time: Clipboard

For both types of interaction, we logged the time taken to complete the data transfer operation. For clipboard data, we recorded the time between the copy/cut operation and the first corresponding paste operation.

The mean time between the two sets of operations was 13.7sec (s.d. 18.5sec), with the distribution of these durations shown in Figure 4. From these observations we see that the majority of clipboard operation pairs are chronologically close (76.7% under 10sec), indicating participants usually locate the destination window/application of the operation quickly, rather than retaining the data on the clipboard while completing other interactions.

5.3.2. Transfer Time: Drag-and-Drop

For drag-and-drop data, we recorded the time taken to complete the operation, from the time the mouse button was depressed, until when it was released. The mean drag-and-drop action completion time was 6.0sec (s.d. 5.5sec), see Figure 5. Despite the large deviation between participants, we observed a far greater percent of actions occurring in the 0–2sec range (47.4% compared to 15.1% in copy-and-paste). This shorter interaction time (when compared to clipboard operations) is likely caused by the direct nature of drag-and-drop; it does not allow the user to perform other actions mid-operation and requires continual mouse input to retain the 'dragged' data. Drag-and-drops may also occur over shorter distances (for instance when positioning items in a Presentation Editor) meaning shorter interaction durations.

5.3.3. Data Type: Clipboard

We recorded the type of data transferred (and its size) for clipboard operations. Of the 31,567 total clipboard operations, 28,939 (92%) were text and 2,628 (8%) were files. An average across participants of 63% textual and 9% file operations occurred within the same application (s.d. 17.3% and 10.6% respectively); 27% textual and 2% file operations occurred between different applications (s.d. 13.8% and 5.9%).

Of the textual data transferred by participants, 61.3% (s.d. 14.0) of actions within documents were on strings less than 50 characters in length; between documents this fell to 44.0% (s.d. 18.4%), see Figure 6.

For the transfer of files, where the logger could report this information (71% of occurrences), we recorded whether one or multiple files were transferred. 74% of the time, one file was transferred, while 26% of transfers involved multiple files.

5.4. Patterns of Use

Many of the clipboard actions occurred chronologically close together: as an average across participants, 44.9% of actions (s.d. 11.1%) occurred as a pair, 8.6% (s.d. 4.0%) as a triplet, and 11.6% (s.d. 6.3%) as four actions with less than 10seconds between each action. In total, 46.8% of actions (s.d. 13.8%) occurred as part of 'bursty' clipboard interaction that involved *more than* two interactions. We also observed several extreme examples: P8 continuously pasted text 869 times (interleaved by three cut/copy actions) into one application; while P5 recorded 213 continuous copy and paste actions transferring text between multiple documents.

5.5. Methods of Initiation

We recorded the initiation method for all clipboard operations (Figure 1). Participants can initiate clipboard operations by using the mouse/trackpad cursor (menus, buttons, or context menus, 30.0%, s.d. 32.5%) or through keyboard shortcuts (70.0%, s.d. 32.5%).

To understand if the input method changes mid-operation, we analysed the method of initiation for source and destination clipboard actions. We observed a small increase in the average use of the keyboard between source and destination operation (2.6%) and the same drop between source and destination use of cursor-based methods. This shows stable use of the same input modality across a full clipboard operation.

Further, we analysed for drag-and-drop whether the available pointing device (trackpad or mouse) influenced the frequency of use of this interaction. On average, mouse users drag-and-dropped 0.53 times per interaction hour, whereas trackpad users performed 0.85 drag-and-drop operations per interaction hour.

6. Discussion

This paper characterised participant use of the clipboard and drag-and-drop across a longitudinal log study of 90 days. We analysed five categories of participant interactions: (1) frequency of data transfer; (2) location and source/destination application; (3) transfer time, type and size; (4) patterns of use and; (5) methods of initiation. We discuss the key findings with implications for desktop UI design.

6.1. Clipboard use is More Common than Drag-and-Drop

Participants utilised data transfer actions with the clipboard more frequently than drag-and-drop. This is likely due to clipboard actions' increased visibility in user interfaces (toolbar buttons, menu items), the context of use, and the widespread adoption of consistent keyboard shortcuts across applications. Conversely, drag-and-drop is a 'hidden' interaction that was historically inconsistently implemented across applications. The recent widespread adoption of touch-based direct manipulation interfaces however, may support increased awareness of this style of data transfer. User interface designers should consider: (1) enhancing current clipboard systems ahead of drag-and-drop systems due to their extensive use; (2) providing better awareness of drag-and-drop interactions, especially when they can lead to more efficient user behaviour.

6.2. Object-Based Interfaces Encourage Drag-and-Drop

The use of the clipboard and drag-and-drop differs across applications. This is likely due to the different interface visualisations and paradigms, and their intended use. For example, more

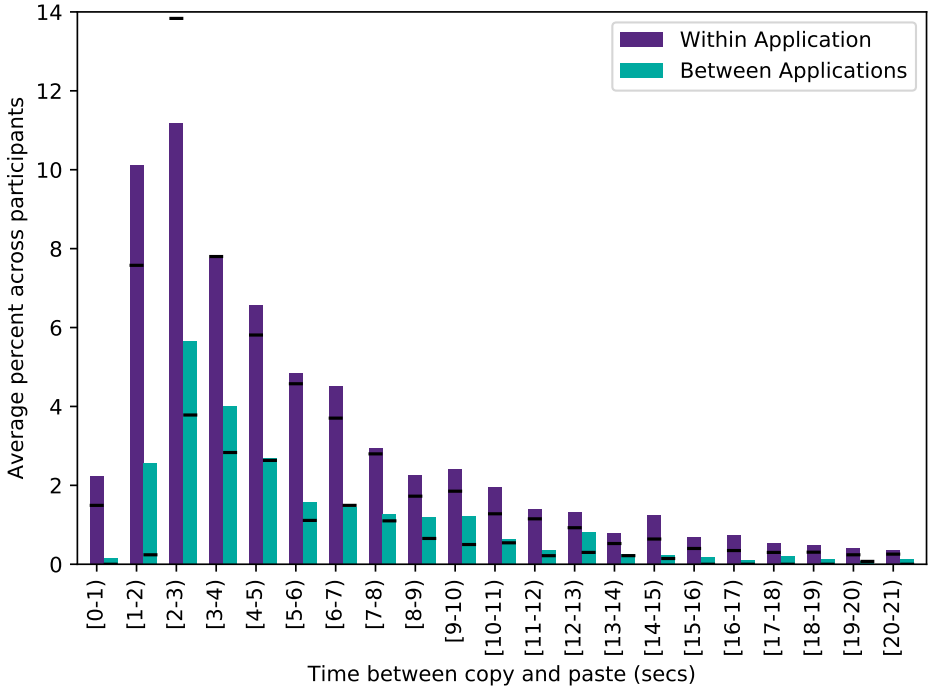

Figure 4: Distribution of times between cut/copy and paste action (average percent across participants, black horizontal marks indicate median values).

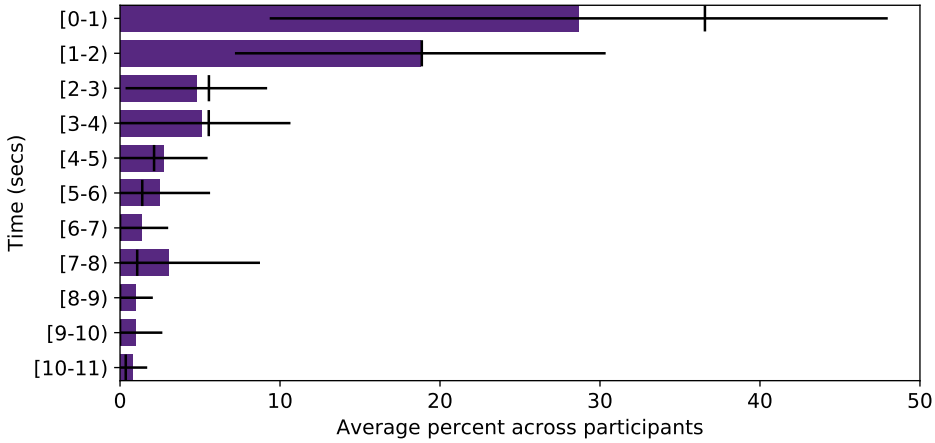

Figure 5: Distribution of drag-and-drop data transfer times (shown as average percent across participants, black vertical marks indicate median values).

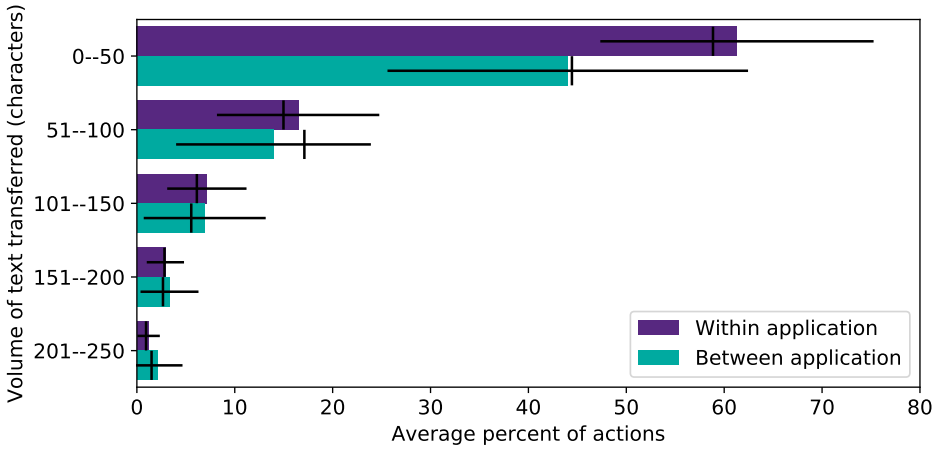

Figure 6: Length of clipboard text transferred by number of characters (truncated at 250 characters for clarity, black vertical marks indicate median values).

drag-and-drop operations occur in Windows Explorer than clipboard operations—this interface encourages direct manipulation of objects (files). Google Chrome however, has less drag-and-drop operations and more clipboard interaction—in this case there are fewer obvious objects to manipulate. While there is no clear 'best' data transfer technique, designers should ensure that they appropriately cater for the transfer technique most likely to be implied by the interface style—in visual-centric interfaces users will expect drag-and-drop.

6.3. Data Transfers are Quick and Involve Text or Files

Overall, we found that most data-transfer operations involve text or a single file and that the average time between the copy/cut action and corresponding paste action (or drag-and-drop duration) was short (<20sec in the majority of cases). This indicates that participants were either pre-aware of the destination or could quickly locate the destination object/window.

However, for between window clipboard operations we observed, on average, 1.6 window switches between the copy/cut action and the corresponding paste operation. Ideally, this would be 1.0 switches (i.e. users immediately find the destination window), possibly indicating a need for improved understanding of the relationship between different windows/applications during data transfer. In the future, window-switching tools could better support users' understanding of the relationship between windows—if data is copied-and-pasted between two windows this forms an implicit underlying relationship, potentially helping them later re-identify that pair of windows.

6.4. Clipboard Operations are Triggered by the Keyboard

Most clipboard operations are initiated through keyboard shortcuts, rather than a pointing device input. Within our study group this indicates a good transition to expert behaviour. Future work should analyse the navigation actions that occur between the cut/copy and paste actions with a view to further increasing user efficiency.

6.5. Drag-and-drop Operations are More Likely on Trackpads

When comparing different pointing devices, we observed an average of 0.85 operations per interaction hour for trackpad users and 0.53 operations per interaction hour for mouse users. We postulate that the increased use of the trackpad could be due to the spatial immediacy of the trackpad compared to the mouse, requiring users to move their hands less distance to trigger the drag. Further, and anecdotally, it is easier to accidentally drag an object with the trackpad than it is with the mouse: our study was unable to differentiate between intended and accidental actions.

6.6. Comparison to Previous Characterisations

To understand how user behaviour has changed since previous reports of data transfer, we briefly compare some of the core statistics that were reported across papers:

Frequency of Data Transfer: In their study of programmers using cut-and-paste, Kim [2] report 16 instances per hour of use; in comparison, we saw only 5.3 actions per hour. This difference is due to the study setups—Kim studied people for a short duration in their work environment performing particular tasks; our study captures a wide range of both work and leisure use.

Within and Between Application Transfer: Our data shows an identical ratio of within to between application data transfers as that reported by Stolee [1]. Our data saw 69.5% of actions within a document, while Stolee reports 70%—this proportion has remained constant in the time between studies.

Most Frequent Applications for Data Transfer: Stolee reports Word processors (26%), web browsers (23%), email clients (19%), spreadsheets (18%) and IDEs (5%) as the most frequent applications for data transfer. In comparison, we saw IDEs, presentation editors, spreadsheet editors, email clients, and text editors as the most frequent users of data transfer tools (Figure 3). These differences

are due to the group of participants in the studies—for example, a sample with non-programmers would clearly not use IDEs.

Amount of Unused Data: Our data shows 1.6 unused cut/copy actions per hour out of a total of 5.3 clipboard operations per hour (giving 30% unused); Stolee reports 19% of their cut/copy actions are unused. Although we observed a slightly higher rate of 'unuse', this behaviour has not changed significantly.

Length of Text Data Transfers: We observed 61% of text transfer actions were for short strings (<50 characters); Kim observed 74% of copy-and-paste actions consisting of "a single line such as a variable name, a type name or a method name"—likely equivalent to our 50 characters. This indicates that Kim's findings here also apply to the broader scope of computer use.

6.7. From Characterisation to Tool (Re-)Design

This characterisation provides the empirical foundation for researchers and practitioners to design or re-design existing data transfer techniques. We envision a multitude of different applications of this data, but highlight here some promising avenues for investigation:

Support for Progression to Expert Behaviour: We observed several participants primarily using the menus for clipboard interaction (P10, P16, P01, see Figure 1) and many participants with no or infrequent use of drag-and-drop. This provides strong evidence of the lack of *intermodal* progression [19] from novice to expert controls and should be used as motivation for designers to examine users' routes to progression and how these might be improved. Further, we identified core patterns of use that have the potential to drive automation tools that might make 'data transfer suggestions' analogous to 'search suggestions'.

Awareness of Clipboard Content: We observed regular unused cut or copy actions (1.6 per interaction hour, see Figure 2). While we lack the contextual information to understand why (incorrect cut/copy, using cut as delete, forgotten action) there is scope to provide increased awareness of clipboard content, either continually or just in those circumstances when an unused cut/copy is (about to be) overwritten. These could take the form of ambient or contextually relevant notifications that may help to increase interaction efficiency.

Cross-screen Data Transfer: We observed few examples of data transfer across displays (15% for clipboard and 0.9% for drag-and-drop). This may be indicative of participants' utilisation of multiple displays (e.g. different contexts on different displays) but may also indicate the need for better tools to support multi-display interaction (especially for drag-and-drop). These observations are inline with early qualitative understanding of multi-monitor use [20], where users reported that their second monitor was used for secondary activities, peripheral information awareness, and for easy access to other resources. Additional study that includes finer-grained task understanding and spatial understanding of windows and activities is required to fully understand how novel mechanisms can improve cross-screen data transfer.

Relationships between Applications: There is significant scope to support users in understanding the relationship between different application windows—data transfer indicates some kind of relationship. We observed 1.6 window switches between the cut/copy and paste action in between-application data transfer. By helping the user to understand these relationships we can potentially decrease the time they take to find the destination window and therefore increase their interaction efficiency. Data-transfer information could be used as an input for predictive algorithms for supporting window/application searching and revisitation (e.g. alternatives to traditional recency or frequency-based approaches) or visualisations.

6.8. Limitations

While the 90-day data collection should expose even rarely used interactions, such a study still includes some limitations. Although MultiLog allows vast data collection from many sources, the Windows APIs it relies on to report clipboard and drag-and-drop events can occasionally miss or report incorrect data. Where detected, these were removed from the data set before analysis commenced. Our analysis makes some assumptions, for example, that a drag-and-drop action is not longer than 60 seconds. However, this may remove some, very long interactions, such as drag-autoscroll-drop which we were unable to detect. Further, we are unable to determine whether a drag-and-drop action results in the user returning an object to its starting position. To ensure full user privacy, we did not record when MultiLog was paused (and unpaused) during the study. However, this means that we do not know how many hours of user interaction was not recorded and how this might change the reported data.

We feel 90 days (3 months) is a sufficient period to gain an accurate overview of user habits, a longer study (up to a year) would help us understand if users' behaviour evolves over time. The participant cohort was fairly balanced in terms of gender and age, but for greater insight, future studies should include an even more diverse user-base. Many of our participants had technical backgrounds—this user group *may* exhibit more 'expert' behaviour (e.g. greater proficiency with shortcuts) than participants from other backgrounds; this is also worthy of consideration in future studies.

Although we collected quantitative log-based data, we could have also collected other useful data such as screen-recordings of users at work, interviews or used direct observation methods. These methods would provide a more qualitative-based data set, allowing us to better explore users' context, thoughts and feelings in addition to the log data.

7. Conclusion

This paper characterised user behaviour with desktop data transfer tools (clipboard, drag-and-drop) over a 90-day log study. We identify and present a set of new trends surrounding the frequency of data transfer, location and application source/destinations, transfer time, type and size of data transfer, patterns of use, and methods of initiation. We found that within application data transfer operations are more frequent than between applications. We observed a short time and low number of window switches between the copy/cut and paste operations. The results and discussion provide a starting point for further research and the continued (re-)design and development of data-transfer support tools.

8. References

[1] K. T. Stolee, S. Elbaum, G. Rothermel, Revealing the copy and paste habits of end users, in: 2009 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)(VLHCC), volume 00, pp. 59–66.

[2] M. Kim, L. Bergman, T. Lau, D. Notkin, An ethnographic study of copy and paste programming practices in oopl, in: International Symposium on Empirical Software Engineering, ISESE '04, IEEE, pp. 83–92.

[3] O. Chapuis, N. Roussel, Copy-and-paste between overlapping windows, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '07, ACM, New York, NY, USA, 2007, pp. 201–210.

[4] G. Faure, O. Chapuis, N. Roussel, Power tools for copying and moving: Useful stuff for your desktop, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '09, ACM, New York, NY, USA, 2009, pp. 1675–1678.

[5] J. Woodruff, J. Alexander, Multilog: A tool for the control and output merging of multiple logging applications, Behavior research methods 48 (2016) 1296–1307.

[6] C. Brown, The Essential Guide to Flex 2 with ActionScript 3.0, Apress, 2007.

[7] M. Kobayashi, T. Igarashi, Boomerang: Suspendable drag-and-drop interactions based on a throw-and-catch metaphor, in: Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, UIST '07, ACM, New York, NY, USA, 2007, pp. 187–190.

[8] P. Baudisch, E. Cutrell, D. Robbins, M. Czerwinski, P. Tandler, B. Bederson, A. Zierlinger, et al., Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch-and pen-operated systems, in: Proceedings of INTERACT, volume 3, pp. 57–64.

[9] S. A. Brewster, Sonically-enhanced drag and drop, in: International Conference on Auditory Display, ICAD '98, Georgia Institute of Technology, 1998.

[10] C.-H. Shih, H.-C. Huang, Y.-K. Liao, C.-T. Shih, M.-S. Chiang, An automatic drag-and-drop assistive program developed to assistive people with developmental disabilities to improve drag-and-drop efficiency, Research in Developmental Disabilities 31 (2010) 416–425.

[11] B. Shneiderman, H. Kang, Direct annotation: A drag-and-drop strategy for labeling photos, in: B. B. Bederson, B. Shneiderman (Eds.), The Craft of Information Visualization, Interactive Technologies, Morgan Kaufmann, San Francisco, 2003, pp. 58–65.

[12] U. Kukreja, W. E. Stevenson, F. E. Ritter, Rui: Recording user input from interfaces under windows and mac os x, Behavior Research Methods 38 (2006) 656–659.

[13] Microsoft Corporation, PersonalVibe, http://research.microsoft.com/en-us/downloads/0ea12e49-8b29-4930-b380-a5a00872d229/, 2009. Last accessed: 18/09/2018.

[14] S. Tak, A. Cockburn, Window watcher: A visualisation tool for understanding windowing activities, in: Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7, OZCHI '09, ACM, New York, NY, USA, 2009, pp. 241–248.

[15] Microsoft Corporation, Microsoft Problem Steps Recorder, http://windows.microsoft.com/en-gb/windows7/how-do-i-use-problem-steps-recorder, 2016. Last accessed: 18/09/2018.

[16] J. Alexander, A. Cockburn, An empirical characterisation of electronic document navigation, in: Proceedings of Graphics Interface 2008, GI '08, Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 2008, pp. 123–130.

[17] J. Alexander, A. Cockburn, R. Lobb, AppMonitor: A Tool for Recording User Actions in Unmodified Windows Applications, Behavior Research Methods 40 (2008) 413–421.

[18] Microsoft Corporation, UI Automation, https://docs.microsoft.com/en-gb/windows/desktop/WinAuto/entry-uiauto-win32, 2018. Last accessed: 19/09/2018.

[19] A. Cockburn, C. Gutwin, J. Scarr, S. Malacria, Supporting novice to expert transitions in user interfaces, ACM Comput. Surv. 47 (2014) 31:1–31:36.

[20] J. Grudin, Partitioning digital worlds: Focal and peripheral awareness in multiple monitor use, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '01, ACM, New York, NY, USA, 2001, pp. 458–465.