

Understanding and improving cross- application interaction in desktop computing



Jonathan Woodruff

School of Computing and Communications

Lancaster University

This dissertation is submitted for the degree of

Master of Philosophy

August 2020

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This thesis is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. Many of the ideas contained within this thesis are as a result of discussion with my supervisor, Dr. Jason Alexander. This thesis contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Jonathan Woodruff

2020

Contributing Publications

Woodruff, J., Alexander, J. “MultiLog: a tool for the control and output merging of multiple logging applications” *Behavior Research Methods* 2015.

Woodruff, J., Alexander, J. “Data transfer: A Longitudinal Analysis of Clipboard and Drag-and-Drop use in Desktop Applications” *International Journal of Human-Computer Studies* 2019.

Acknowledgements

I would like to thank the Engineering and Physical Sciences Research Council (EPSRC) for providing the funding which allowed this research to take place. I'd like to extend a big thank you to my supervisor and mentor Jason Alexander, who has provided constant support and advice throughout the MPhil process. Thank you, Jason, for all your time advising, proof-reading papers and providing research and personal advice. I certainly could not have achieved this without you.

I would like to thank my wife Abigail Woodruff for all her emotional support throughout the process of my MPhil. I would also like to thank my father, Barry Woodruff, for the extensive advice and assistance with programming aspects of the MPhil, along with financial assistance. Thank you to my mother, Sue Dickinson, for her mental and financial help. Thank you also to my grandparents, Bob and Pat Dickinson, for always being there for advice and assistance, particularly to Grandad for ensuring I was always on track and Grandma for her expert proof-reading of my work! Thank you to my sister, Sophie Woodruff, and my uncle and aunt, Giles and Jenny Mayor, for caring and assisting me throughout the MPhil process and to my cousin, Thomas, for providing distractions and many a football game in the back garden. Thank you to the rest of my family for your continued support throughout the MPhil process.

Thank you also to my best friends, Joe Porter and Stephen Owens for offers of assistance and advice throughout the process. Thank you to Stephen for the many lunches held on campus which allowed me to get out of the office for a while and re-focus. Also, thank you Stephen for your advice and assistance with programming issues and MPhil conceptual ideas that proved very useful.

Thank you to members of my research group - Kiel Long, Faisal Taher, Aluna Everitt, Miriam Sturdee, Adam Blaney and the wider School of Computing & Communications department for all their assistance throughout my MPhil.

Abstract

Cross-application interaction on modern computer operating systems is becoming more prevalent as users begin to work more dynamically and utilise more applications simultaneously to complete a task. Such tasks could consist of writing a research paper, developing software or preparing a presentation. These tasks all require the use of multiple applications to complete. However, the interaction between these applications is still poorly understood.

At present the research community does not have an extensive overview of the ways in which users work with multiple applications to complete tasks. Previous research mainly focusses on interaction within windows which lacks the cross-application element. Data flow between applications is also not fully understood. Research into a more conceptual, higher level approach to working, with a view to understanding how interaction and communication between these applications aids users when completing tasks, is required.

This work has two main aims: 1) To understand how users utilise the open application set to complete tasks and 2) To improve cross-application interaction for users. Within (1) MultiLog (an extensive logging framework) is presented. The MultiLog system is used to gather data during a 90 day study logging users' interactions with their PCs. This data is then analysed and presented. Within (2) this thesis uses these results to inform the design of QuickFileAccess which aids users when locating folders by dynamically managing the Windows Quick Access list.

The results of the log study enabled previously published results to be updated. New results in the area of data transfer, covered a deep understanding of not only how users employed the Windows clipboard to transfer data between applications, but also how

participants utilised the “drag-and-drop” facility for data transfer. Results informed the design and development of the QuickFileAccess tool.

Table of Contents

Declaration	ii
Acknowledgements	iv
Abstract	vi
List of Tables	xiii
List of Figures	xiv
Chapter 1: Introduction	1
1.1 Research Questions	4
1.2 Research Contributions	5
1.3 Methodology	6
1.4 Thesis Outline	8
Chapter 2: Related Work	10
2.1 Activity-Based Computing/Cross-Application Interaction	11
2.2 User Modelling	14
2.3 Recommender Systems	15
2.4 Logging Software	18
2.1.1 Low-Level Loggers	18
2.1.2 High-Level Loggers	19
2.4.2.1 Application-Specific Loggers	19
2.4.2.2 Generic Loggers	20
2.1.3 Screen Recorders	20
2.1.4 Log-Merging Software	20
2.1.5 Summary	21
2.5 Window Switching	21
2.2.1 Window Switching Tools, Methods, and Patterns	22
2.2.2 Desktop Screen Setup	22
2.2.3 Window Management Tools	24
2.2.4 Novel Window Switching Techniques	24

2.2.5	Summary	25
2.6	Data Transfer	25
2.3.1	Clipboard (Cut, Copy, Paste)	25
2.3.2	Drag-and-drop	26
2.3.3	Summary	27
2.7	Summary of Related Work.....	27
Chapter 3: MultiLog		29
3.1	Introduction	30
3.2	Description of MultiLog.....	32
3.2.1	Researcher Mode.....	32
3.2.2	Deployment Mode	33
3.2.3	Deployment.....	34
3.2.4	Summary	34
3.3	MultiLog Architecture.....	34
3.3.1	Command-Line Loggers	36
3.3.2	Stand-Alone Loggers	36
3.3.3	TCP/UDP Loggers	37
3.3.4	Logger Summary	37
3.3.5	Set up and Use	37
3.3.6	MultiLog's Parsing of Log Files.....	38
3.4	MultiLog Validation.....	38
3.4.1	Compatibility with Existing Logging Systems	39
3.4.2	Data Integrity	44
3.4.3	Resource Consumption	46
3.4.4	Pilot Testing	48
3.4.5	Discussion	48
3.4.6	Summary	49
Chapter 4: Log Study		50
4.1	Purpose of Log Study	50
4.2	Types of Data to be Collected	51
4.3	Types of Data Not to be Collected	52
4.4	Loggers Required	53
4.3.1	Window Switching & Windows Explorer File Directories	53
4.3.2	Mouse Clicks & Keyboard Entry.....	55
4.3.3	Windows Clipboard	55

4.3.4	Drag-and-drop.....	56
4.5	Length of Study.....	57
4.6	Participant Overview.....	57
4.7	Start of Study.....	58
4.8	End of Study.....	59
4.9	Qualitative Feedback.....	59
Chapter 5: Understanding User Interactions.....		60
5.1	Window Switching.....	60
5.1.1	Results.....	62
5.1.1.1	Window Switching Methods.....	62
5.1.1.2	Position & Size of Windows.....	66
5.1.1.3	Number & State of Windows/Applications at Switch Time.....	71
5.2	Data Transfer.....	76
5.4.1	Data Analysis & Results.....	77
5.2.2	Results.....	77
5.2.2.1	Frequency of Data Transfer.....	77
5.2.2.2	On-Screen Location & Source/Destination Applications.....	80
5.2.2.3	Transfer Time, Type & Size.....	83
5.2.2.4	Methods of Initiation.....	86
5.3	File Directory Access.....	87
5.4	Summary of Insights.....	88
5.4.1	Window Switching.....	88
5.4.2	Data Transfer.....	90
5.4.3	File Directory Access.....	91
5.4.4	Summary.....	92
Chapter 6: QuickFileAccess.....		94
6.1	Justification of Approach.....	94
6.2	Context.....	96
6.3	QuickFileAccess Concept.....	98
6.4	Description.....	98
6.5	Algorithm Development.....	99
6.6	Final Algorithm.....	101
6.7	Results Presentation UI.....	102
6.8	Algorithm Limitations.....	103
6.9	Beta Testing.....	103

6.10	Pilot Study	103
6.11	Pilot Study Results	105
6.11.1	Frequency of Items Clicked in the Quick Access list	105
6.11.2	Frequency of Items Added to the Quick Access list.....	106
6.11.3	Frequency of Items Removed from the Quick Access list.....	106
6.11.4	Content of the Quick Access list.....	106
6.11.5	Qualitative Feedback from Users.....	108
6.11.6	Summary of Results	109
6.12	Pilot Study Discussion	110
6.13	Pilot Study Strengths/Weaknesses	111
Chapter 7: Discussion and Future Work		113
7.1	Progress on Research Questions	114
7.1.1	How can we collect data on multi-application use in unmodified desktop environments?.....	114
7.1.2	How and when does multi-application use occur in desktop environments?.....	115
7.1.3	How can we improve the efficiency of interactions in multi-application desktop environments?	116
7.2	Lessons for Practitioners	116
7.3	Research Generalisability.....	118
7.3.1	MultiLog	118
7.3.2	Analysis.....	119
7.3.3	QuickFileAccess	119
7.4	Cross-Application Interaction into the Future.....	120
7.5	Limitations	120
7.5.1	Operating System.....	121
7.5.2	Participant Makeup	121
7.5.3	Participant Makeup Reflection.....	122
7.5.4	Contextual Information	123
7.5.5	Data in Isolation	124
7.5.6	Contextual Information	124
7.6	Future Work	124
7.6.1	Better Understanding Interactions of Users	125
7.6.2	Development of Further Useful Tools/Visualisations	126
Chapter 8: Conclusion.....		127
References.....		130

Appendices.....	137
Appendix 1: Study: Information for Participants.....	138
Appendix 2: Study: Installation Instructions	145
Appendix 3: Study: Demographics Survey.....	148
Appendix 4: QuickFileAccess Pilot Study: Information for Participants.....	153
Appendix 5: QuickFileAccess Pilot Study: Installation Instructions.....	162
Appendix 6: QuickFileAccess Pre-Deployment Survey.....	164
Appendix 7: QuickFileAccess Post-Deployment Survey	174
Appendix 8: MultiLog & QuickFileAccess Software Removal	177
Appendix 9: QuickFileAccess Withdrawal.....	180

List of Tables

Table 1: A list of loggers tested with MultiLog. Y = logger works as expected, N = logger does not work, ! = logger partially works but there are known issues as detailed in the Notes column.	40
Table 2: A list of three loggers (with three different types of output) along with example raw and MultiLog-parsed data.....	43
Table 3: Loggers used for output validation.	45
Table 4: Loggers used for performance validation. All tests were run for a period of 15 minutes apart from the Arduino light sensor test which was run for 30 minutes. The time intervals selected were relatively short because each test requires continuous interaction within that application, something current time constraints did not permit. ‘Interaction’ refers to the PC being utilised for the duration of the test. ‘No interaction’ means the PC was idle for the duration of the test. The tests performed with the Arduino light sensor were performed on a different machine due to hardware constraints. The no interaction figure provides a baseline from which the overheads can be calculated.	47
Table 5: Log study participant overview.	58
Table 6: Types of window switch.	63

List of Figures

Figure 1: MultiLog’s researcher mode with Microsoft PSR, Mozilla Firefox and Tobii Eye Tracker loggers enabled. This allows researchers to see the locations on screen users are looking while performing their everyday tasks.	31
Figure 2: MultiLog architecture.	35
Figure 3 Window switches by type (average across participants).	62
Figure 4: Window switches by type (average across participants) using the Gaylin (1986) categorisation.	63
Figure 5: Window switch initiation method (average across participants).	64
Figure 6: Window switch initiation method by display setup using the Tak et al. (2009) classification.	64
Figure 7: Average time between window switches (minutes).	65
Figure 8: Heat maps for three users’ main window distribution.	67
Figure 9: Heat maps for three users' sub window distribution.	67
Figure 10: Heat maps for three users' dialog window distribution.	68
Figure 11: Heat maps for three users' progress window distribution.	69
Figure 12: Heat maps from Tak (2011) results.	69
Figure 13: Heat maps from multi-screen users.	70
Figure 14: Window switches to the same or different screens for multi-screen users.	70
Figure 15: Percentage average of windows in states (by single- and multi-screen participants).	71
Figure 16: Example of Windows z order.	72
Figure 17: Depth of target window in window stack.	72
Figure 18: Position of the target window in the z-ordering of windows (Tak).	73
Figure 19: Sum total (across participants) of applications switched to.	73
Figure 20: Distribution of windows (top) and applications (bottom) open at switch time.	75
Figure 21: Tak (2011) results showing average number of open windows.	75
Figure 22: Breakdown of clipboard operations by type.	78
Figure 23: Distribution of the number of drag-and-drop operations performed per hour across all participants.	79
Figure 24: Comparison between within and between application data-transfer operations (average across participants).	81
Figure 25: Average within-application (top) and between-application (bottom) clipboard operations per hour. Utilities are applications such as Windows Explorer, Calculator, and Remote Desktop. Not the different horizontal scales.	82
Figure 26: Average drag-and-drop actions per hour by application group.	82
Figure 27: Distribution of times between cut/copy and paste action (average percent across participants). Graph truncated at 20sec for clarity.	84
Figure 28: Distribution of drag-and-drop data transfer time (shown as average percent across participants). Graph truncated at 10sec for clarity.	84
Figure 29: Length of clipboard text transferred by number of characters (data over 500 characters (10%) is not shown for clarity).	85
Figure 30: Number of window switches between copy/cut and paste (between-application data transfer only).	86
Figure 31: Clipboard operations by initiation method.	86

Figure 32: Clipboard operations by initiation and input method.	87
Figure 33: Average drag-and-drop operations per interaction hour by input method.	87
Figure 34: Windows Explorer "This PC" view (Windows 10). Quick Access list is highlighted in red.	96
Figure 35: Windows Explorer Quick Access view (Windows 10).	97
Figure 36: Original "Suggested Folders" Windows Explorer overlay.	102
Figure 37: Breakdown per participant of the number of times items in the Quick Access list were clicked.	105
Figure 38: Breakdown of directories added to the Windows Explorer Quick Access list by QuickFileAccess across participants.	106
Figure 39: Breakdown of sum total number of items displayed in Windows Explorer Quick Access List.	107
Figure 40: Frequency of number of items shown in Windows Explorer Quick Access list when queried.	108
Figure 41: Overview of participant's Quick Access usage prior to pilot study.	108
Figure 42: Participant responses post-pilot study.	109

Chapter 1: Introduction

Most modern computer systems are built on a platform of applications with which the user interacts. These different applications provide different types of functionality depending on which task the user is currently performing. They can be used together with other applications or in isolation to perform a given task.

Cross-application interaction (the use of multiple applications and interactions between them) is an interaction technique required to achieve certain tasks. Most users of computers utilise their PCs to perform a given task (Abela, Staff, & Handschuh, 2010), not to use a specific application. Bihler and Kniesel (2007) state this problem well: “Currently, user interfaces are defined by single applications but workflows may span multiple ones”. Most tasks require the use of multiple applications in tandem in order to accomplish the given task and as such, most users will employ some level of cross-application interaction while working on tasks. The fact most users will use their PCs to perform a given task means research in the cross-application area is extremely important and relevant as it has the ability to affect millions of users worldwide. The cross-application interaction area is not at present fully understood as we do not currently understand how people use applications together and this thesis aims to provide an overview of the ways in which users interact with multiple applications. Research that has been carried out in the area of cross-application interaction so far is not up-to-date and does not include findings on newer interaction techniques or different interaction styles users are performing which are related to cross-application interaction. The most recent papers include Chen, Perrault, Zhao, and Ooi (2014) and Abela et al. (2010) are from 2014 and 2010 respectively. As a result, further research is required into understanding these interactions and ultimately, cross-application interaction.

Work in the area of cross-application interaction has been studied across several sub-fields of HCI, including activity-based computing, user modelling and recommender systems. Activity-based computing states that a user utilises a PC to perform a task, not to use a certain application (Abela et al., 2010). This thesis sits in the area of providing an overview of how cross-application interactions (activity-based computing) are performed on desktop computers. The area of user modelling (Clemmensen, 2004) covers building conceptual understanding of users. This thesis characterises user behaviour based on the results of a longitudinal log study undertaken as part of the research. Recommender systems (Adomavicius & Tuzhilin, 2011) are software tools and techniques providing suggestions for items to be useful to a user. This thesis' QuickFileAccess tool contributes to this area. It is a plug-in for Windows Explorer which recommends relevant directories to the user based on their interactions. This thesis fills a knowledge gap by providing an up to date understanding and then subsequently a tool which aims to improve these cross-application interactions through a deep characterisation of user behaviour and tools to aid users when performing these interactions and some is not recent.

One of the main desktop interactions a user has within the operating system is with windows (Tak et al., 2009). When the user encounters a higher number of open windows (applications can have multiple windows open simultaneously), it can be hard to navigate between windows or to select the window/application required. Finding the correct resource among many can be a time-consuming task and can decrease user efficiency. This level of inefficiency with the interaction of a high number of windows prompted research into the efficiency of window switching systems and possible improvements. Such research has dominated the investigations into the cross-application interaction area of Human-Computer Interaction (HCI) in past publications.

Various studies were conducted into the ways in which participants switched windows and the methods used to do so, for example Tak et al. (2009). As a result of this research and analysis, the research community developed various tools which aided the user when accessing and switching between windows and made these tasks less laborious. Many of these tools are described in the Related Work section of this thesis. Much of this work is not recent and has not been carried out with the newest operating systems currently available. This thesis aims to provide new findings in the areas of window switching, data transfer and file directory management and update previous findings in

the area of window switching alongside analysing how trends have changed and/or have stayed consistent over time.

With the increased number of applications becoming available to computer users, interaction techniques using multiple applications has started to increase. While this increase has taken place, research into this area of HCI has been undertaken but is still fairly sparse and requires further attention in order to properly understand user habits and ways in which efficiency can be improved. In order to improve these cross-application interactions users perform, the research first needed to understand how they are utilising applications currently. This thesis examines the many interaction techniques in the areas of (1) window interaction including window switching, size and usage, (2) data transfer (including clipboard operations and drag-and-drop actions in particular) and (3) file directory management including which directories are accessed. It is hoped that the findings of this thesis can help inform the designs of operating systems in the future making it easier for users to perform cross-application tasks and therefore increasing their efficiency.

To increase the knowledge in the cross-application area of HCI and in order to achieve the increase in user efficiency, this thesis aims to understand and improve the ways in which users interact with their computers in a cross-application manner. In order to understand this behaviour, a longitudinal log-based study was conducted with 17 participants for a period of three months. In order to collect the data of interest, a tool had to be developed that allowed the recording of user behaviour. MultiLog is a tool publicly available for the collection of log data (such as window switches, clipboard operations, keyboard input and mouse clicks) and merging from multiple pre-existing logging applications. This bespoke software allowed the gathering of a range of data in areas that have not been explored in depth before. Information on window switching, Windows clipboard usage, “drag-and-drop” activity and file directories accessed was collected as part of the study.

The findings from the study give an insight into how users are utilising the open set of applications to complete tasks, how they interact with various different windows, how they transfer data between applications and which file directories they are accessing.

The findings also allowed the development of QuickFileAccess which is an extension of the MultiLog framework providing feedback to the user based on live analysis of log

data collected through the main MultiLog software. Relevant file directories are shown to the user through the modification of the existing Windows Explorer Quick Access list to present relevant directories based on an algorithm developed as part of the research which was based on the longitudinal log study analysis described in Chapter 5.

1.1 Research Questions

This work sits in the area of extending our current understanding of cross-application interaction and has two main goals: (1) to understand and (2) to improve cross-application interaction for users of desktop computers. In order to develop tools which attempt to improve user efficiency, a deep understanding of how users are performing cross-application interactions and the relationships between applications and those that exist between applications had to be sought. This thesis aims to answer the following three high level research questions:

- 1 How can we collect data on multi-application use in unmodified desktop environments?
- 2 How and when does multi-application use occur in desktop environments?
- 3 How can we improve the efficiency of interactions in multi-application desktop environments?

The above research questions provide a high-level overview of the research to be conducted in the area of cross-application interaction. These research questions are highly relevant to the area of cross-application interaction because of Activity-based computing which states that a user utilises a PC to perform a task, not to use a certain application (Abela et al., 2010). As such, the ways in which PCs are used to perform these tasks is highly relevant to the area of HCI in general as it has the potential to increase user efficiency (research question 3) if a detailed understanding (research question 2) can be gathered by data collection (research question 1). As such, these research questions are highly relevant and grounded in the area of improving interactions users perform on their computers which in turn will help the HCI community understand and improve these interactions.

1.2 Research Contributions

This thesis makes five primary contributions to the research community:

- 1 A review of related work in the areas of window switching, data transfer and cross-application interaction. This makes an additional contribution to the area of understanding user interaction in these areas by updating and comparing new results to previous findings from the desktop interaction community.
- 2 The development of MultiLog. MultiLog is a research logging tool which combines the output and control of multiple pre-existing logging applications. It can be used in a “plug and play” nature where existing loggers are “plugged in” through an easy-to-use interface. MultiLog makes a contribution to the logging software area of HCI by providing a framework which researchers can use to gather a powerful set of data with little configuration time for non-expert users.
- 3 A 90 day longitudinal log study undertaken to collect data on user interactions using the MultiLog logging framework. This contributes an updated extensive log study to aid the HCI community in understanding various user interactions.
- 4 A review and characterisation of the ways in which users interact with applications on their computer in a cross-application way. Using the data gathered using the MultiLog tool, results are analysed and an insight into user behaviour is provided. This contributes to the areas of window switching, data transfer and file access. Data from window switching results updates previous findings by others and new findings on data transfer and file access provide new insights into how these systems are utilised currently.
- 5 A new tool QuickFileAccess which aids users when performing tasks requiring the use of multiple applications to increase their efficiency by

making available to them a list of relevant file directories based on live analysis of their interactions recorded by the MultiLog framework. This provides a contribution to the area of interaction design and recommender systems by pilot testing a tool which recommends file directories based on an algorithm designed by the earlier collected data.

1.3 Methodology

The methodology section sets out the methods used during the MPhil to answer the research questions set out in previous sections. It also provides justification of the choice of research methodology and relevant advantages and disadvantages of the methods chosen and utilised throughout the research.

The overall approach of this thesis is to adopt a log-based strategy to collect data regarding the interactions users are performing on their PCs and to use this data to attempt to improve these interactions to ultimately increase user efficiency. The approach to answer the first research question is to investigate whether applications currently exist which facilitate the type of data collection required through a longitudinal log study. If no current solution can be found, an application which facilitates the required data collection will be designed and developed. It will then be used to conduct a quantitative log study to gather data about user interaction across applications. This method of gathering data on computer usage has been used in various previous studies successfully including Tak et al. (2009), Hutchings, Smith, Meyers, Czerwinski, and Robertson (2004) and Gaylin (1986). These papers provide a detailed characterisation of user behaviour by collection and analysis of log data. This work follows a similar procedure to these past studies, effectively re-employing their successful approach. The advantages of the longitudinal log approach are that accurate interaction data can be gathered at a fast rate over an extended period of time. These types of studies also allow researchers to establish patterns/trends over a period of time. Disadvantages of longitudinal log studies are that they require a fairly large number of participants to get a good understanding of user interaction. Other approaches such as video recording or qualitative approaches require input from users who may change certain behaviours if they have time to think about them or when the observation is intrusive. Video recording is also time-intensive and qualitative approaches are highly subjective. It was decided a quantitative approach would give the most valuable data

and consequently, logging software will be required for this stage and research into whether a logging architecture that fits these needs currently exists will be undertaken and, if necessary, it will be developed as part of the research. A longitudinal approach is best in this case as it helps to understand how users interact with their PCs over a period of time, allowing the identification of trends in interactions and the comparison of results with past longitudinal studies.

The approach to answer the second research question is to systematically analyse the data collected during the execution of the longitudinal log study. The characterisation of behaviour is the best approach in these circumstances as it allows the understanding of exactly how users utilise their PCs currently, which will hopefully enable the research to improve users' efficiency by aiding cross-application interaction. On completion of the study, the resultant data will be analysed in detail to characterise users' activity with a view to identifying task-oriented interactions. Data, including interactions with applications, files, e-mails, web pages and various other standard computer features, will be analysed in order to try to ascertain whether users are utilising their computers in a task-based way. Directed questions of the data were posed in order to form this data characterisation.

The final question would take the approach of experimental computer science to create a semi-automatic tool which will attempt to help users work across applications more efficiently. The empirical results of the previous analysis were studied, and an area of improvement selected for the semi-automatic tool to focus on improving user efficiency. The semi-automatic tool would aid users by presenting relevant directories through the Windows Explorer Quick Access list through an algorithm devised as a result of the previous research questions, aiming ultimately to increase user efficiency by decreasing the amount of time spent locating directories/files. Visualisations can also aid users and can complement these tools as they could see the cross-application interactions occurring on their computer which could prompt the improved use of interactions with applications in order to increase their efficiency. The development of a tool has a direct advantage to the end-user over other methods (such as visualisations or other means) in that it would work with the user directly and could possibly automate some of the processes of working in a cross-application way, something visualisations can only show the user and other approaches would not have such a direct involvement. This would in turn attempt to decrease the user's workload, something it is believed

other methods would not do substantially. A semi-automatic tool was developed as a prototype which managed the Windows Explorer Quick Access list based on directories it deemed relevant from an algorithm devised from the answers to the previous research questions in this thesis. A small pilot study designed in order to evaluate the tool was conducted and results presented.

1.4 Thesis Outline

The remainder of this thesis is divided into seven chapters: Related Work, MultiLog, Log Study, Understanding User Interactions, QuickFileAccess, Discussion and Future work and Conclusion.

The Related Work chapter explores previous research relating to the MPhil in the areas of logging software, window usage, data transfer and cross-application interaction. A full overview of available logging software is presented, along with the disadvantages of using existing systems over the creation of a new universal logging framework. The main drawback of using such a system is the lack of a high-level overview of user behaviour. Pre-existing loggers focus on more niche data collection, for example window switching or keystrokes.

The MultiLog chapter describes the newly-developed MultiLog logging framework in detail. It explains why the development of new logging software was required for the studies this research wanted to undertake. The design and development process of MultiLog is then described fully, which is followed by an in-depth explanation as to how MultiLog works. The ways in which the software controls and merges the output of most pre-existing logging applications is explained. The strengths and drawbacks of the software are described and a thorough conclusion given.

Within the Log Study chapter, a description is given surrounding the longitudinal log-based study that was implemented using the MultiLog software to better understand cross-application user behaviour. The participant recruitment process, demographics and length of study are described fully along with the relevant ethical considerations, data protection and the security precautions taken to keep individual user data safe.

The Understanding User Interactions chapter describes the findings of the longitudinal log-based study in detail. Three major analyses form this chapter: Window Switching,

Data Transfer and File Directory Access. Within the Window Switching section this thesis presents results surrounding the ways in which participants interacted with windows with emphasis on switching methods, size and position of windows, window re-use and z-order (depth). The Data Transfer section analyses the use of the Windows clipboard and the “drag-and-drop” method of transferring data within and between applications. The methods participants used to interact with these systems were logged, along with the duration of the operations and which applications these occurred between. Within the File Directory Access analysis this thesis presents results regarding how users utilise and access certain file directories within their system. This analysis helped inform the design of the QuickFileAccess tool.

The QuickFileAccess chapter describes the extension to the MultiLog logging framework and how it operates. It explains how the development of the tool was undertaken and its features in detail. It also describes how the extension was deployed for evaluation in the form of a small pilot study. Results of the deployment are presented both in quantitative (from log data) and qualitative (from user feedback) forms.

Within the Discussion and Future Work chapter extensive discussion is presented surrounding the MPhil including all previous sections of this thesis. The MultiLog logging software, log study, the log analysis and the QuickFileAccess tool are all discussed including limitations of such systems/work. A comprehensive Future Work section provides an overview on possible directions of future research, including new user studies and further tools and visualisations which could be developed to aid the computer end-user.

Chapter 2: Related Work

This chapter provides a review of related work in the core areas of the research. Related work covers research on (1) activity-based computing/cross-application interaction, (2) user modelling, (3) recommender systems, (4) logging software, (5) window switching and (6) data transfer. Within (1) an overview of current and past cross-application work is provided so that this research can better understand the area of cross-application interaction and understand the gap in the research community which this thesis hopes to fill with the new tool QuickFileAccess. This work aims to answer the third research question on improving the efficiency of interactions. In (2) user modelling is explored and a selection of relevant work on user models is presented aiding the second research question on how and when multi-application use occurs in desktop environments (through this thesis' user characterisation). In (3) background into recommender systems is presented which paves the way for this thesis' recommender system, QuickFileAccess with an aim to increase user efficiency (research question three). Within (4) an overview into pre-existing logging systems for longitudinal data collection is provided and relates to the first research question – surrounding understanding how data can be collected on multi-application use in unmodified desktop environments. In (5) research into various areas surrounding window use such as window switching is presented. This allows the analysis and understanding of data collected using the logging system. Within (6) background information is presented on the areas of the Windows clipboard and data transfer between applications. Both (5) and (6) allow us to understand and answer the research question surrounding understanding how and when multi-application use occurs in desktop environments. Related work within (4) links to the first research question on how data can be collected on multi-application use.

2.1 Activity-Based Computing/Cross-Application Interaction

Activity-based computing is the concept that “the basic computational unit is no longer the file (e.g. a document) or the application (e.g. MSWord) but the activity of a user”. As such, it is now widely understood that users utilise PCs for the purpose of completing a given task and not to use a specific application (Abela et al., 2010). These tasks often require multiple applications. These applications typically do not have an easy way for the user to interact between them (Bihler & Kniesel, 2007). An increasing number of tasks performed on computers now require at least two or more applications to be used in order to complete a particular task. For example writing a research paper may require Microsoft Word, an internet browser with multiple tabs and several reference papers contained within PDF files. Bardram, Bunde-Pedersen, and Soegaard (2006) explain in depth the background relating to activity-based computing in their CHI paper. The paper presents activity-based computing, a core concept for this thesis. The Windows taskbar is replaced with an activity bar to better support users habits for working by task and not by application. This system was designed and implemented on Windows XP.

Some work in this area has focussed on improving cross-application interactions in general. One such paper that supports this cross-application interaction is Bihler and Kniesel (2007) who focus on identifying the problem of disruptive cross-application workflows in ubiquitous computing and they also propose a dynamic user interface fusion to support seamless cross-application workflows. Further, they identify technical challenges for interface fusion not solved by existing approaches and review supporting techniques possibly applicable in the area. The authors present a solution which contains interface “merging”. For example, instead of simply showing “Person A calling” with “Answer” or “Reject” buttons the interface shows “Person A Calling” with “Answer” and “Reject” buttons but also has additional contextual information from other types of applications such as the last e-mail sent to/from the person calling and making visible any previous notes that were made during the last call the user had with the person in question. This provides additional contextual information to the user which may be useful in deciding whether to answer the call and may also provide an insight as to the purpose and content of the call.

Another paper in the area of general cross-application interaction is by Fenstermacher and Ginsburg (2002). In this research a new framework that monitors higher-level events to learn how people access, create, and modify information rather than how they use applications. Application use is monitored extensively and inter-application communications such as the Windows Clipboard is also monitored. Various high level events were logged from applications such as Internet Explorer, Microsoft Word and the Windows Clipboard. The software developed uses Windows APIs and the COM events system to log the relevant events. This paper shows that people are working in task-based and not application-based ways and as such, the research and commercial community should be doing more to increase the efficiency of the user when using multiple applications to complete a given task.

Bihler and Kniesel (2007) also recognise the issue of general cross-application interaction. They state the problem well: “Currently, user interfaces are defined by single applications but workflows may span multiple ones”. They describe in detail this disconnect between applications and workflow and focussed on how to identify the user’s current task and how user interfaces can be re-organised across multiple concurrently used applications. The paper goes on to explain how application features are segmented, grouped together due to a given task context and displayed using “Dynamic User Element Fusion” (as mentioned in the earlier paper). Throughout the paper, the authors address the problem of supporting cross-application context with dynamic user interface fusion which is a technique to merge interface parts from independent applications or service providers.

Other research focusses more on cross-application interaction between certain types of applications. For example, Budzik, Hammond, and Birnbaum (2001) claim that user interactions with productivity applications, for example, word processors and web browsers, provide rich contextual information that can be leveraged to support just-in-time access to task-relevant information. In their paper they present *Watson* which is a system that gathers contextual information repositories related to the task at hand, as well as process explicit requests in the context of this task. The questions they address surround how information access systems benefit from knowing more about a user and how systems can be built in this way. An evaluation of this research is also undertaken to assess whether this actually assists the user in any way. The *Watson* tool analyses, for example, a paper a researcher is writing and returns relevant documents, news

images/video and for sale items relevant to the current task. This may assist the user in providing information that the user does not have to manually search for. This thesis aims to produce a tool which also aids users when performing certain tasks on their PCs in order to make it easier and quicker for them to perform a particular task.

A tool that focusses on cross-application interaction within web browsers is *Letizia* (Lieberman, 1995) which is an “agent that assists web browsing” and is an interface that assists users when browsing the internet by tracking behaviour and anticipating the interests of users by analysing this collected data. It automates a browsing strategy consisting of a best-first search augmented by heuristics inferring user interest from browsing behaviour. An example of *Letizia*'s use is also explained whereby a user browses the web for articles on Artificial Intelligence. A user focusses in on articles on Agents by searching for “Agents” in the page or looking at pages referring to Agents in particular. From this browsing behaviour an interest in “Agents” can be derived which helps suggestions in the tool to be presented. Later, the user is browsing personal home pages which may contain a list of their publications. As the user is browsing the list of publications *Letizia* can also browse the list for topics of inferred interest, in this case the “Agents” example from earlier. *Letizia* does not only suggest documents that may be relevant but also explains to a certain extent why the relevant documents were chosen. In this case it noticed a keyword from a previous exploration and in the other case, a comparison was made to a document that also appeared in the list returned by the bibliography search.

The research community has also examined cross-application mobile interaction with the introduction of the tool *BezelCopy* (Chen et al., 2014) which focussed on mobile device cross-application copy and paste. The *BezelCopy* system is a system for copy and pasting across applications on smartphones. In order to perform the copy and paste the user performs a bezel gesture to select the desired sentences in text to copy. The selected text appears on a new panel to enable fast and precise selection and a list of icons appear on the bottom of the screen which can be selected as the target application to paste the text. The text is then appended to the end of the document in the target application. This work shows how important it is to enable cross-application interaction on mobile devices as well as desktop devices which is something of interest for future work.

This thesis aims to update findings/analyses on this cross-application interaction/activity-based computing, particularly those interaction techniques which are now more prominent than in the past due to the ways in which people are now completing given tasks on PCs and not using specific applications (Abela et al., 2010). In order to characterise how users are currently using their PCs in an activity-based way, various cross-application interactions such as window switching, data transfer and file directory access analyses will be presented to further the knowledge in the area of cross-application interaction/activity-based computing and inform the design for a new recommender system developed as part of the thesis.

2.2 User Modelling

User Modelling is the concept of building a conceptual understanding of users. A key challenge of the HCI area is to make systems/interfaces more usable. Specifically presenting the right things at the right times in the right ways (Fischer, 2001). System designers have the challenging job for designing one piece of software for millions of users as if it were designed on a per-user basis. The article reviews the objectives, progress and unfulfilled hopes that have occurred in the past with regard to user modelling. Biswas and Robinson (2010) present a brief survey of different user modelling techniques used in HCI. They classify existing models into different categories which include: (1) The GOMS family of models, (2) Cognitive Architectures, (3) Grammar-based models and (4) Application Specific models. One such relevant paper which presents user modelling is Abela et al. (2010). This paper proposes a Task-Based User Model (TBUM) which acts as a user's mental model for each task by automatically tracking, relating and organising resources associated with the task at hand. The TBUM system can be used to restore a Task Space (set of applications required for a given task) when an interruption occurs or when the task is suspended by the user.

The paper "Four approaches to user modelling – a qualitative research interview study of HCI professions' practice" (Clemmensen, 2004) presents four types of experienced HCI professionals who are interviewed about their ways of describing users. The results of the interviews with the HCI professionals provided information on four effective approaches to user modelling in practice. These are: (1) the "analyst", (2) the

“designer”, (3) the “programmer” and (4) the “bridge builder”. Within (1) the analyst approach conceptualises the user as one or more abstract personality. (2) models users as characters in need of information with the purpose of creating an image of the users in the head of customers. (3) conceptualises the user in data to be shown at the computer display and (4) describes the user in terms of his or her tasks, with the purpose of evaluating prototypes and removing usability problems. This work both aids this thesis when characterising user behaviour gathered during the longitudinal log study and is also relevant to the work on the third research question of this thesis surrounding improving these characterised interactions through the development of the QuickFileAccess tool.

May, Barnard, and Blandford (1993) show how structural descriptions of interface designs can be used to model user tasks, visual interface objects and screen layouts. The ideas presented within the work build “approximate” descriptions of the cognitive activity underlying task performance in interactions. They further show how such descriptions provide a firm basis both for predicting performance in experimental settings, such as learning task structures or searching for icons and also for analysing and understanding user behaviour with dynamic display implementations, such as the hypertext example.

This thesis aims to further research the area of characterising user behaviour by analysing data collected as part of a longitudinal log study on window switching, data transfer and file directory access. This in turn may strengthen user models to better understand cross-application interactions users are employing. Previous work (Fischer, 2001) states that user modelling techniques based on logged user data (which is what this thesis presents) can support the organisation-wide learning of HFAs (High Functionality Applications). The link between user modelling and characterising user behaviour through log studies is further explained in Linton, Joy, and Schaefer (1999) where models are developed on collected log data from user interactions.

2.3 Recommender Systems

Recommender Systems are software tools and techniques providing suggestions for items that might be useful to a user. They do this by using algorithms or information gathered to inform the list of recommended options. Recommender systems have

succeeded in domains as diverse as movies, news articles, web pages, and wines (Cosley, Lam, Albert, Konstan, & Riedl, 2003). The idea is that if user behaviour is tracked and analysed successfully, appropriate recommendations can be made to users based on this behaviour.

Cosley et al. (2003) study two aspects of recommender system interfaces that may affect users' opinions: the rating scale and the display of predictions at the time users rate items. Results found that showing predictions when users rate movies changes their ratings although it was not clear how long that change of opinion lasts. It was also found that user satisfaction suffered when they manipulated ratings, probably due to lower accuracy. They state that much of the accuracy of recommendations in these types of systems has been solved by well-tuned algorithms.

Lops, De Gemmis, and Semeraro (2011) provides an overview of content-based recommender systems, with the aim of imposing a degree of order on the diversity of the different aspects involved in their design and implementation. The chapter is split into three subsections. The first explores the basic concepts and terminology of content-based recommender systems and is presented along with their high level architecture and advantages and disadvantages. The second covers a review of the state of the art of systems adopted in several application domains, by thoroughly describing both classical and advanced techniques for learning user profiles. Finally, trends and future research are discussed.

Perugini, Gonçalves, and Fox (2004) state that recommender systems attempt to reduce information overload and retain customers by selecting a subset of relevant items based on user preferences. They state that these systems have traditionally been studied from a content-based filtering vs. collaborative design perspective. They state that all recommender systems make connections among people and thus should be surveyed from such a perspective. Information overload has also prompted research and work into recommender systems to show more relevant information to users.

Jannach, Zanker, Ge, and Gröning (2012) reviewed and classified recent research in recommender systems both in the field of Computer Science and Information Systems. Existing trends are identified along with open issues and possible directions for future research. Various collaborative filtering techniques are used in the paper. The literature review indicates the importance of recommender systems in these areas and the authors

found Computer Science researchers focus more on algorithms while Information Systems researchers are more interested in the systems-perspective and the effects of recommender systems on the users. They state that both areas can inform each other for mutual benefit.

Fitchett and Cockburn (2012) introduce AccessRank which is an algorithm that predicts revisitation rates in many contexts including file accesses (something relevant to this thesis). The system uses many sources of information to generate its recommendations including recency, frequency, temporal clustering, and time of day. The AccessRank system performed better than other such systems as it was considered to be more stable. Results and real-world examples of how real world applications could use AccessRank is explained.

Fitchett, Cockburn, and Gutwin (2013) explore the problem surrounding improving navigation-based file retrieval. They state that “navigating through a file hierarchy is one of the most common methods for accessing files”. However they also state this method can be slow and repetitive. Some new algorithms have the ability to improve this process but it is unknown how best to present this to users. The paper presents three design goals: (1) Minimise the time spent at each hierarchical level en route to the target file, (2) reduce the number of levels traversed by providing shortcuts and (3) promote rehearsal of the retrieval mechanics to facilitate expertise. These goals were designed to improve user performance when accessing files. The interfaces presented are designed to assist with hierarchical traversal, ultimately assisting users to be able to learn the location of relevant files.

Further work will be conducted by this thesis in the area of recommender systems by providing the QuickFileAccess tool which recommends relevant file directories based on cross-application interactions the user has performed being analysed in real time and presented in the Windows Explorer “Quick Access” list. This work will further research on recommender systems by providing an up-to-date recommender system for file directories.

2.4 Logging Software

The first of the research questions explores the better understanding of how users employ multiple desktop applications in order to complete tasks. The first stage of this section of research is to gather data surrounding users' interactions. Logging of users' actions is valuable in understanding how people utilise applications and interfaces on their computers. The resultant output can then be used to inform the future (re-)design of operating systems. This section reviews logging systems in general, a range of existing loggers, and previous work into merging output from different loggers. The following sections detail the types of logging applications reviewed during fulfilling the first research question.

Logging is useful both within and beyond the HCI community. An example of a field outside the HCI community where logging is helpful to researchers is Interactive Information Retrieval (IIR) which also uses collected log data to analyse and draw conclusions from users' behaviour. Kelly and Sugimoto (2013) provide a deep overview of IIR evaluation studies and the use of logging as a data collection technique used within them.

2.1.1 Low-Level Loggers

Low-level loggers typically just record basic input actions such as mouse movements and keystrokes. Examples include *Actual Keylogger Software* (Actual Keylogger, 2014), *REFOG Free Keylogger* (REFOG, 2014) and *A Stealthy GPU-based Keylogger* (Ladakis, Koromilas, Vasiliadis, Polychronakis, & Ioannidis, 2013). Such loggers allow researchers to analyse, for example, how people use certain keyboard shortcuts (Peres, Tamborello, Fleetwood, Chung, & Paige-Smith, 2004) or how fast they type (Kinkead, 1975). The biggest disadvantage of low-level loggers is their lack of context. Without knowing which buttons or menu items are clicked or which textboxes text is entered into researchers can only make general statements about user behaviour.

2.1.2 High-Level Loggers

High-level loggers are either application-specific or generic and provide additional contextual information to that of low-level loggers. They are either targeted at a particular software application or more generally at an operating system.

2.4.2.1 Application-Specific Loggers

Application-specific loggers are sometimes developed and shipped as part of a software package and often encourage users to ‘opt-in’ to product improvement programmes (e.g. the (Adobe Customer Improvement Program, 2014) and the (Microsoft Customer Experience Improvement Program, 2009)). Other loggers, such as *Microsoft Outlook Logging* (Microsoft Corporation, 2003) and *Microsoft Visual Studio Logging* (Microsoft Visual Studio, 2014) support logging for diagnostic and troubleshooting purposes. Log files are saved to disk and can be sent to the manufacturer for inspection. Examples from the research community include *AppMonitor*, a logger to document detailed events performed within Microsoft Word and Adobe Reader (Alexander, Cockburn, & Lobb, 2008) and *OpenOffice.org Interceptor*, an application which uses a hybrid technique to log events inside OpenOffice.org applications (Dostál & Eichler, 2011).

An increasing number of applications include some form of high-level logging software. Browsers such as Google Chrome (Google Chrome, 2014) contain an in-built logger that records a list of websites visited; such logging is invoked via the command line. Other applications, such as those from the Mozilla family ((Mozilla Firefox) & (Mozilla Thunderbird)), also support this level of logging. Unfortunately, these loggers must be manually started and stopped via the command line and work on extracting relevant lines from the output (depending on the research being implemented) needs to be conducted, which non-expert users may struggle with.

Application-specific loggers’ tight integration with specific software means they can log detailed contextual events within the application but cannot provide insights into the external context.

2.4.2.2 Generic Loggers

Generic loggers gather information from basic operating system events such as window switches and detecting when applications are started or stopped. Examples include *RUI* (Kukreja, Stevenson, & Ritter, 2006), *PyLogger* (Tak & Cockburn, 2009) and *VibeLog* (Oliver, Smith, Thakkar, & Surendran, 2006) which monitors the windows a user switches between. The tool *Morae* (TechSmith, 2015) is also relevant and collects data for market research purposes on usability and other types of pre-release product testing. While these cover a broad spectrum of basic actions users undertake during operating system-level interaction the level of detail differs between loggers.

Generic loggers have the advantage of providing external contextual information as basic events across the whole computer system are monitored and they are not attached directly to a specific application.

2.1.3 Screen Recorders

Screen-recorders collect a series of screenshots or video feeds of the user's desktop providing full details of their actions both within and between applications. They are often complemented with a low-level logging system in order to provide accurate timing information that would be tedious to extract from the video stream (e.g. *Wintective* (Wintective, 2014)). The main limitation with screen-recorders is their resource requirements: recording for extended periods of time can be resource intensive for both CPU usage and storage. Examples include *CamStudio* (CamStudio, 2014), *Rylstim* (Rylstim Screen Recorder, 2014) and *Ezvid* (Ezvid, 2014) which collect video feeds of the user's desktop.

2.1.4 Log-Merging Software

Merging the data collected from multiple loggers is not new. IBM have in the past worked in the area of autonomic computing where they devised the *Generic Adapter Logging Toolkit* (Grabarnik, Salahshour, Subramanian, & Ma, 2004) which provides a framework for transforming event-based system information into a standard format. Although it transforms event data into a generic format, it does not control the applications producing the log files, nor does it cater for researchers with only basic

computer knowledge as an adapter (software module) has to be written or, as seen in further work (Math, Seetha, Kulkarni, & Yardi, 2009), a regex defined for each file which is performance-intensive. *A Generic Log Analyser* (Shahzad, 2013) has also been developed as part of a master's project but instead focuses on tracking down issues in the mobile telecommunications area, it has no control over the applications, and requires an XML schema to be provided in order to parse the files.

Other commercial applications such as *Log Monitor* (Log Monitor, 2014) and *Tiny Log Monitor* (Tiny Log Monitor, 2014) bring logger output together into one application. *Log Monitor* allows the user to open multiple log files and watch them for changes while *Tiny Log Monitor* supports the same functionality but with the addition of adding regex patterns to format the output. In both these applications the output from different log files is not saved, merged or uploaded and they do not remove individual log lines, all of which are key features of the logging application developed in this thesis, MultiLog.

2.1.5 Summary

This thesis presents MultiLog which is a dynamic tool for the merging of pre-existing loggers. The MultiLog tool provides researchers with a framework for logging user interactions which is easily configurable to non-expert users and works with many pre-existing logging applications. This is a contribution of this thesis to the area of logging software.

2.5 Window Switching

The second research question covers better understanding of how users employ multiple desktop applications together in order to complete tasks. As part of this section of research data on window switching on Windows desktop computers was gathered. Window switching is a fundamental interaction performed by computer users and is a required action which allows access to applications and resources. Within this section of the related work four areas are explored: (1) window switching tools, methods, and patterns; (2) desktop display setups; (3) window management tools and; (4) novel window switching techniques.

2.2.1 Window Switching Tools, Methods, and Patterns

Window switching methods have evolved over time. Gaylin (1986) found through direct observation that “cycling through windows” (which is similar to the present day Alt + Tab) was the most frequent switching method (50.8%). Gaylin (1986) also found the mean number of windows open at one time was 3.7 and that modifying window set up (size and location) is much more common immediately after the window opens than later in its lifetime.

Hutchings et al. (2004) collected similar window switching data (through logging on Windows XP) and hypothesised that participants utilised the general window click method (clicking directly inside the destination window) more than a taskbar selection, these results indicated a shift in primary switching method since Gaylin (1986). The techniques for window switching have also changed between the two studies possibly explaining the differing results (by an increase in clicking in the destination window and a decrease in keyboard-based methods). Hutchings et al. (2004) also noted that the taskbar has potential “usability problems” due to its location and configuration. During this study windows were typically active for 20.9 seconds.

A more recent study of window interaction (Tak et al., 2009) analysed switching techniques in a longitudinal data set (Alt + Tab, general window click, and taskbar). A measure of these three methods across users with single- and multi-screens showed that multi-screen users are more likely to use direct click and less likely to use the taskbar for window switches. Single-screen users are more likely to use Alt + Tab than multiple screen users. Tak (2011) report a window re-visitation rate of 82%; 80% of all window switches being to 37% of windows and 80% of all application switches being to approximately 2-11 applications. Tak (2011) also reports the average number of open windows at any particular time was 8.5.

2.2.2 Desktop Screen Setup

There is anecdotal evidence to suggest that desktop users have an increased workload and number of tasks to complete and are increasingly turning to larger and/or a greater number of screens to aid their completion. Previous research including Bi and

Balakrishnan (2009) and Czerwinski et al. (2003) compared large screens to single- or multiple-screen set-ups. Kang and Stasko (2008) look at the use of window interactions within users of single- and multiple-screen systems and Grudin (2001) investigates secondary screen use on desktop computers.

In the early 2000s single screens were considered sufficient to conduct tasks in standard software (Mackinlay & Royer, 2004) and when present, secondary screens were used for secondary tasks (Grudin, 2001). Mackinlay and Royer (2004) undertook a 3 month longitudinal log study and results showed that most skilled workers could have a “reasonable pattern of window activity” with one screen. They also found that windows typically fill the screen (are in maximised configuration). Grudin (2001) backs up claims made by Mackinlay and Royer (2004) showing that “secondary screens are generally used for secondary activities related to the principal tasks”, leaving the majority of tasks to be completed on the primary screen. They noted that a secondary screen “improves efficiency in ways that may be difficult to measure”.

Kang and Stasko (2008) validated previous findings (Mackinlay & Royer, 2004) and (Grudin, 2001) by presenting results which showed that participants who used two screens performed tasks faster (and with less workload) than those who used one. Participants expressed a subjective preference for the multiple-screen set up over the single-screen.

Results in this area identify trends in which users become more efficient with more than one screen but use the secondary screen(s) for secondary activities with most primary tasks being undertaken on the primary screen.

Larger screens have become more frequently used in recent years (Tak, 2011). Czerwinski et al. (2003) investigated performance benefits of using larger screens compared to the traditional screen set up. Results showed that the average task completion time was lower on larger screens. Bi and Balakrishnan (2009) also undertook a study that focused on how large displays (16 foot x 6 foot) are used. Their results indicated that the general preference amongst users was for the large displays as it allows them to interact with multi-window systems and perform “rich information tasks” as well as enhancing users’ awareness of peripheral applications.

2.2.3 Window Management Tools

Screen space is a valuable asset (Kamba, Elson, Harpold, Stamper, & Sukaviriya, 1996) among users but it is not practical to display all windows simultaneously (Tan, Meyers, & Czerwinski, 2004). While viewing all windows at the same time is possible there may be a large amount of space between relevant information. Tan describes an interaction technique that allows a user to create a copy of a particular region from an open window thus organising the position of windows to possibly aid switching techniques. Window management tools aid the organisation and acquisition of windows. Henderson and Card (1986) highlights the issue of windows competing for space and presents the *Rooms* system to aid this competition. The *Rooms* system splits the desktop into multiple virtual desktops (rooms). Other systems such as Voida, Mynatt, and Edwards (2007) and Hutchings and Stasko (2002) developed systems that aimed to address issues with window location and screen space. Voida's *Gionata* system leverages the spatial organisation of a user's work area by constructing activities (tasks) which are a collection of tools/windows/applications. Hutchings and Stasko's *QuickSpace* system focuses on a set of fundamental operations that allow users to quickly allocate space to a window using operations that have a direct correspondence to physical actions. Other systems such as Houben, Vermeulen, Luyten, and Coninx (2012) and Dragunov et al. (2005) help the user organise their windows in task-groups which, in turn, aid them when switching between tasks.

2.2.4 Novel Window Switching Techniques

Novel window switching techniques have employed eye-gaze to facilitate faster and subjectively preferred interaction. Kumar, Paepcke, and Winograd (2007) produced a technique for switching between windows using a combination of keyboard and eye gaze. When evaluating this system they compared it to the traditional methods of Alt + Tab and the taskbar. Taskbar selection was found to be more accurate, though, and the author's *EyeExpose* system beat Alt + Tab and taskbar in every category. Fono and Vertegaal (2005) present a system whereby users switch between windows with their eyes. Results showed that users subjectively preferred switching techniques that involved some form of eye interaction.

2.2.5 Summary

This thesis provides updated analyses on window switching including not only interactions which were logged by previous research, but also explores new areas which have not been studied before in order to give a detailed overview into the ways in which window switching is occurring in the present day.

2.6 Data Transfer

Data transfer is relevant to the second research question of this thesis - understanding how and when multi-application use occurs in desktop environments. This thesis gathered data on data transfer between applications. Related work in the data transfer area covers the ways in which data is moved both within and between different applications. The main methods of data transfer are the Windows clipboard and “drag-and-drop” operations. Relevant related work falls in two key areas: (1) Clipboard (copy-and-paste) and; (2) Drag-and-drop.

2.3.1 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, Elbaum, and Rothermel (2009) details early work in understanding user interactions with the clipboard. They studied 15 participants’ clipboard use over four weeks, reporting the type of applications and a breakdown of source and destination applications. 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 application.

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.

Miryung, Bergman, Lau, and Notkin (2004) 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.

2.3.2 Drag-and-drop

A drag-and-drop operation occurs when a user depresses the left mouse button on a draggable object, moves the mouse to the destination location and releases the mouse button to “drop” the object into the destination window/file.

Research into drag-and-drop has primarily focused on directions for the future and its development and use in web-based applications (Drop (2007)). This work notes that drag-and-drop is a frequently used feature of modern operating systems and that there has recently been an increase in web-based drag-and-drop to simplify web services and aid users when interacting with websites.

Further work by Kobayashi and Igarashi (2007), Brewster (1998) and Shih, Huang, Liao, Shih, and Chiang (2010) have explored ways of enhancing the drag-and-drop experience for end users. Kobayashi and Igarashi (2007) present a system that allows users to suspend and resume drag-and-drop operations based on a throw-and-catch metaphor. Brewster (1998) explored whether the addition of non-speech sounds could increase the usability of drag-and-drop while Shih et al. (2010) provided an automatic assistive program to improve the efficiency of drag-and-drop for users with developmental disabilities.

The next generation of drag-and-drop systems (Shneiderman & Kang, 2000) and Jia, Sun, Tang, and Shum (2006) focus on repetitive or complex tasks, where user efficiency is increased by simplifying the interaction to include drag-and-drop actions. These applications aid users with image composition and photograph labelling.

2.3.3 Summary

This thesis furthers work in the area of data transfer by providing the first such study to analyse behaviour in the areas of clipboard and drag-and-drop. The MultiLog logging framework will be used to gather data surrounding these areas and it will be fully analysed, providing the research community with a deep understanding of how the clipboard and drag-and-drop actions are performed, something not currently available.

2.7 Summary of Related Work

Related work covered past and present work relating to the research questions. The chapter provided an in-depth review of related work in the areas of (1) activity-based computing/cross-application interaction, (2) user modelling, (3) recommender systems, (4) logging software, (5) window switching and (6) data transfer. Current logging software does not allow the vast collection of the types of data in the detail required, which opens a gap in the current research leading us to be able to develop the tool MultiLog which hopes to fill this gap. The window switching related work currently available is not recent and so a fresh study on usage of windows on desktop computers is another area of importance to this research. This thesis will provide an up-to-date overview of window usage. Users can now use new techniques to control/switch between windows, something that is new to the latest Windows operating systems. Data transfer has not been studied in detail recently and so this thesis will gather an in-depth understanding surrounding two types of data transfer: the Windows clipboard and drag-and-drop usage. This will help us understand the ways in which users utilise multiple applications to complete tasks (by showing the flow of data between them). This thesis will then present QuickFileAccess, a new tool designed to manage the Windows Explorer Quick Access list to aid users to locate folders more quickly. In summary, this work aims to fill these research gaps by using the newly created logging framework

MultiLog to collect data from users which will be analysed. This analysis will form the basis of the design for the QuickFileAccess tool which aims to increase user efficiency by reducing the time spent locating required files/directories. The tool will also contribute to the area of recommender systems by providing a new system for recommending relevant file directories to the user. All of the above research will aid the core understanding of activity-based computing/cross-application interaction by providing analyses into various cross-application interaction techniques utilised by users and the QuickFileAccess tool to recommend relevant file directories based on cross-application usage.

Chapter 3: MultiLog

The first of the three research questions seeks to understand how data can be collected on multi-application use in desktop environments. This question prompted research to ascertain whether an application which would facilitate this already existed. Findings showed that an application with the exact requirements needed to answer the second research question was not available. As such, MultiLog was developed.

Research into currently available logging applications was conducted and although there is a vast selection of applications in the research and commercial markets, there was no single logging application that would collect the level of information required for the study. However, certain logging applications did provide partial functionality. For example, the Microsoft PSR system provided feedback on user mouse clicks and keyboard entry and the Google Chrome and Mozilla Firefox loggers provided detailed information on which web pages were accessed from the machine. These loggers provided extremely detailed information about one aspect of interaction. As such, it was decided to make use of all these existing logging applications, but a merged, universal log database/file would be easier for analysis of the data. As a result, the MultiLog tool was created. MultiLog controls, gathers, and combines the output, on-the-fly, from existing research and commercial logging applications or “loggers”, presenting results in one universal format. The loggers mentioned above were “plugged-in” to MultiLog and additional loggers (such as to log window switches and drag-and-drop actions) were developed as plug-ins to MultiLog. This chapter describes the MultiLog software in detail. The MultiLog software allows us to gather the required data for research question (1), how can we collect data on multi-application use in unmodified desktop environments.

3.1 Introduction

Client-side logging software runs on a computer to automatically gather data on a user's interactions. This is now a standard method for recording user actions to understand behaviour and improve future interface design as it provides complete, accurate, and machine processable data that catalogues interactions. These loggers may be built into an application, be installed as an extension to an application, or run independent of any application. Typically they are focused on a particular aspect of user behaviour, such as web-browsing patterns (Montgomery & Faloutsos, 2001), window-switching habits (Oliver et al., 2006), or navigation preferences (Juvina & Oostendorp, 2004). Increasingly, longitudinal log analyses are also being used to inform the design of new interface artefacts: Alexander, Cockburn, Fitchett, Gutwin, and Greenberg (2009) design of their *Footprints Scrollbar* from log analysis of within-document re-visitation, while Tak et al. (2009) used longitudinal log data to inform window-switcher design. Hutchings et al. (2004) used the *VibeLog* software to log UI events in order to assess desktop complexity in single and multiple-monitor users. Unfortunately, as loggers typically monitor specific behaviours, researchers can find re-use difficult, as existing systems often do not capture all applicable actions. However, while multiple different loggers in combination can provide the required dataset, their configuration and management are time-consuming and combining their output is difficult.

In order to collect the precise data required for the study, MultiLog was created. MultiLog (Figure 1) is a tool that simultaneously controls, gathers, standardises, and merges the output from pre-existing logging applications. Further, it supports both technical and non-technical experimenters in the deployment of longitudinal logging-based user studies by managing logger start-up, log filtering and obfuscation followed by securely uploading the relevant log files. By easily running multiple logging systems MultiLog also encourages the re-use of pre-existing loggers.

The power of MultiLog stems from its ability to combine output from any pre-existing logging application (providing it has timestamp data) and automate the output combination in near real-time. This enables researchers to understand native PC use in detail by logging actions within multiple different applications (e.g. combining an email logger and file system logger would allow detailed inspection of how a user processes email attachments).

Three key aspects of MultiLog were validated to ensure that the log output was accurate and each of the individual loggers was recording the relevant information correctly: (1) That a diverse range of existing logging systems work successfully with MultiLog; (2) That MultiLog maintains data integrity; and (3) That resource consumption is acceptable. First, a check that a wide range of logging applications and external sensor systems were compatible with MultiLog (both in terms of start and stop configuration, and data capture). Second, data integrity checks were performed to ensure MultiLog successfully parsed and recorded all required data from the logging applications. Third, computing resource consumption tests were conducted to ensure MultiLog did not cause performance degradation when monitoring multiple loggers.

MultiLog is available to researchers via www.scc.lancs.ac.uk/~woodrufj/MultiLog. The MultiLog software is free to download and use for non-commercial and research purposes.

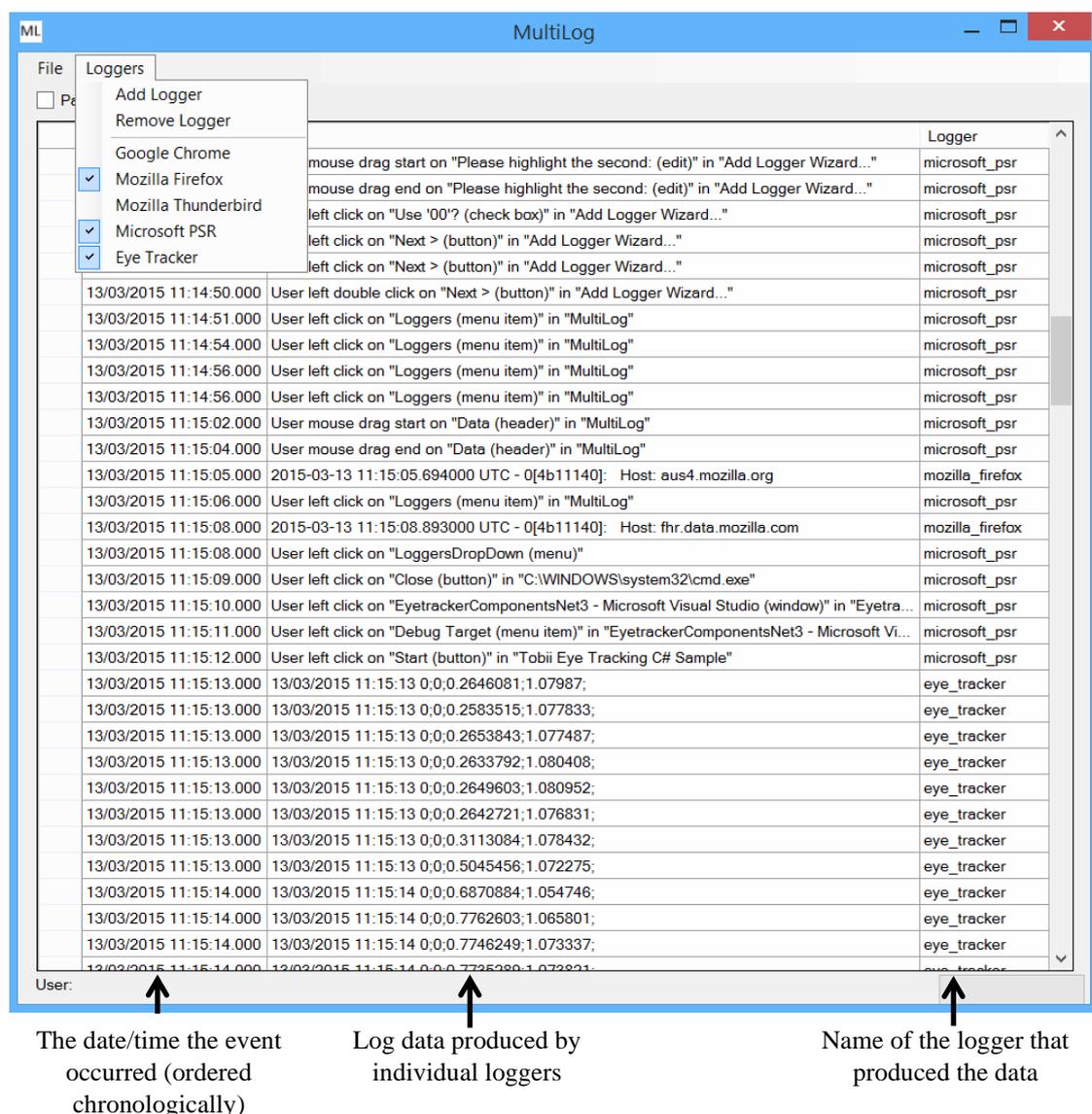


Figure 1: MultiLog's researcher mode with Microsoft PSR, Mozilla Firefox and Tobii Eye Tracker loggers enabled. This allows researchers to see the locations on screen users are looking while performing their everyday tasks.

3.2 Description of MultiLog

MultiLog is a research tool that controls, gathers, filters, and combines the output on-the-fly, from existing research and commercial logging applications. It allows researchers to easily deploy multiple software logging systems to observe user behaviour in either short- or long-term user studies. Automatic log uploading facilitates large-scale data collection.

The system gathers log data on-the-fly: when a logger is enabled, MultiLog actively polls the corresponding log file (or listens on the specified TCP/UDP port) at an interval configurable by the researcher (set to one-minute by default) and checks for updates. If changes are detected (or new data is received on the open TCP/UDP socket) the relevant lines are extracted from the log file, formatted to MultiLog's pre-defined format (as shown in Figure 1) presented in the main interface, and written to an output database.

MultiLog is designed for two groups of users - researchers and study participants - with each having a distinct mode of operation: Researcher Mode and Deployment Mode. Researcher Mode provides the user with full control and configuration ability, while Deployment Mode is intended for user study deployments with settings controlled via a configuration file.

3.2.1 Researcher Mode

By default, MultiLog runs in Researcher Mode where the user sees the full user interface and is able to add and remove loggers, can start and stop loggers, and can view the log output from all currently active loggers, as shown in Figure 1. This mode allows researchers to experiment with logger configurations, examine the combined output from loggers and prepare logging environments for deployment during a user study.

A key feature of MultiLog is its “plug and play” architecture that allows the researcher to “plug-in” any existing logger at any time. MultiLog will work with any existing logging application as long as the researcher can provide the executable name, start and stop commands, the location of the continually-updated log file (or port number if the logger outputs data to a TCP/UDP socket), the position of the timestamp within this output (or the attribute/element that contains the timestamp if the log is in XML format) and an idea of which log lines are required to appear in the output. The “plug and play”

architecture allows even non-technical researchers to quickly configure a series of logging systems. Once configured the researcher can manually start and stop each logger through MultiLog's user interface, sending appropriate signals to the relevant logger.

Researchers can also choose to filter incoming log lines to reduce the amount of data collected. MultiLog supports filtering via line matching to include/exclude text provided by the researcher at configuration time.

The log file polling interval is configurable by the researcher. By default, this is set to one minute which was selected as a result of a trade-off analysis between obtaining real-time data without experiencing degradation in performance. Data received from loggers that output to the TCP/UDP socket is automatically received and processed in real time and thus, the polling interval does not apply to these loggers.

Researchers can “save” the current logger setup (enabled loggers, filters, and polling interval) and generate a configuration file ready to deploy the logger in Deployment Mode.

3.2.2 Deployment Mode

Deployment Mode helps researchers to quickly “roll out” the application to many computers using MultiLog's executable and an editable configuration file. In this mode no interface is displayed and the logger runs “silently” in the user's system tray. The configuration file provides details of each logger to be run (name, executable location, start and stop commands, location of the log file (or port number if the logger outputs data to a TCP/UDP stream), timestamp, and filtering data). If the relevant flag is set inside this file its contents are read by MultiLog on start-up and the relevant loggers are started with MultiLog minimised to the user's PC system tray.

Users can open the interface from the system tray icon, view logged actions, remove individual lines if they do not wish these to be uploaded or pause logging completely. The log lines are stored locally in a database that is automatically uploaded via a secure FTP connection to a server once daily.

In an effort to reduce privacy issues surrounding logging MultiLog can hash the data part of a log line or detect and hash URLs. As an example, when URL hashing is enabled

via the add logger wizard in Researcher Mode the URL `http://www.bbc.co.uk/news/uk/` could appear as `http://www.bbc.co.uk/HGTRFDH`. When enabled, MultiLog detects and hashes the path part of the URL preventing the exact website address from appearing in the output (although identical URLs will hash to the same value). Hashing of the data part of the log line is also set up in the add logger wizard where lines containing certain textual phrases can be hashed.

3.2.3 Deployment

MultiLog saves log data into a local SQLite database that is then uploaded securely (over FTP over SSL) to a server. The local database is then truncated to prevent large amounts of log data accumulating on the user's computer. The researcher configures the connection by providing the address, username, and password of the remote web server. Data can be extracted by non-technical researchers by using MultiLog's re-combination software which combines the output for a given user into a text file.

3.2.4 Summary

The main features of MultiLog are: (1) Two distinct modes of operation for different audiences; (2) Its "plug and play" architecture allowing on-the-fly addition and removal of loggers; (3) On-the-fly gathering, combination and display of logged data; (4) Fully featured Deployment Mode allowing it to start-up and run silently in the user's system tray, allowing user 'pausing' and where necessary removal of log data and hashing to address privacy issues; (5) Log files are securely uploaded to a server on a daily basis via FTP over SSL.

3.3 MultiLog Architecture

MultiLog is written in C# in Microsoft Visual Studio 2013. It has been built to run on Windows 7, Windows 8, Windows 8.1 and Windows 10. MultiLog's high-level architecture is shown in Figure 2.

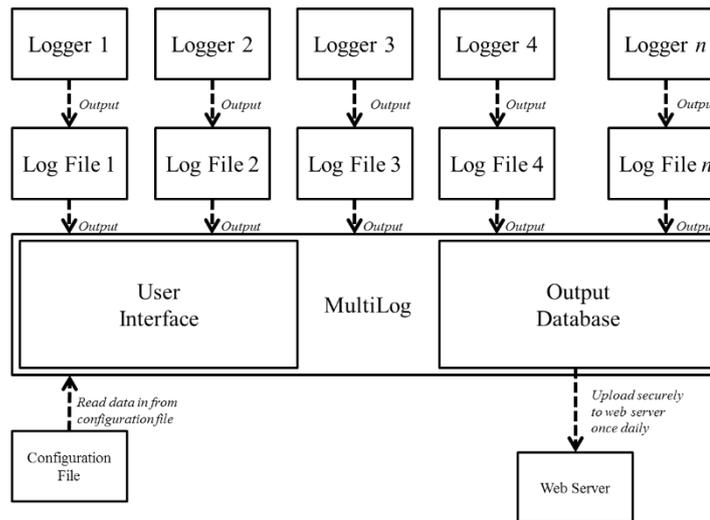


Figure 2: MultiLog architecture.

In its simplest terms, the pre-existing logging applications continue to operate as normal and MultiLog captures their output and merges it into a single interface and output database. MultiLog regularly monitors the logging application’s log file’s contents (by default once a minute, but this is configurable) or listens on the specified TCP/UDP port for a stream of continuous data. When changes are detected or new data is received, it extracts the data from the last line it read to the end of the file or processes the new TCP/UDP data. Internally MultiLog keeps a record of the last position read in each log file. This reduces the overhead incurred by continually reading entire log files (that often become large).

MultiLog can handle three types of logging applications: *Command-Line Loggers*, *Stand-Alone Loggers* and *TCP/UDP Loggers*. Command-line loggers use additional flags to enable internal logging mechanisms on start-up (e.g. Google Chrome, Mozilla Firefox and Mozilla Thunderbird); stand-alone loggers are either dedicated logging applications or are applications that are pre-configured to log events during interaction (e.g. Microsoft PSR); TCP/UDP loggers are systems which send continuous data on specified TCP/UDP sockets. Examples of these type of loggers from the HCI community include eye trackers or EEG systems. The following sections describe how these logger types ‘plug-in’ and interact with MultiLog.

3.3.1 Command-Line Loggers

Many applications now ship with internal logging mechanisms built into the application. These are typically used when trying to trace program bugs and are enabled by setting appropriate command-line flags when starting the application. For example, a user can enable logging in the Google Chrome browser from the command line by running the command `chrome.exe --enable-logging --v=1` in the directory that contains the Chrome executable file. Chrome then generates a continually updated log file of website visits along with other browser events inside the user's home directory. MultiLog comes pre-configured for logging with popular command-line loggers such as Google Chrome, Mozilla Firefox and Mozilla Thunderbird and allows researchers to add additional command-line loggers through the add logger wizard.

The main disadvantage of command-line loggers is that they typically cannot be 'stopped' while the application is running (for example, to stop command-line initiated logging in Google Chrome the user must close and restart the application without the command-line arguments). When a user pauses MultiLog logging from its interface and it is logging from an "unstoppable" application, MultiLog discontinues log polling of that application and keeps track of the last allowable log line. When logging resumes MultiLog only continues log line reporting from the time logging was re-enabled (and does not back-read events that occurred during the paused time). When an individual logger is suspended or removed from MultiLog, MultiLog will provide appropriate warning messages before attempting to close the application.

3.3.2 Stand-Alone Loggers

Stand-Alone loggers are independent applications that record activities within one or more applications or systems, or themselves generate logging information as part of their normal operation. The Microsoft Problem Steps Recorder (PSR) is a typical example that records a wide selection of log events across the whole operating system such as clicks and menu selections, key-presses and shortcuts. MultiLog can typically start and stop external loggers without interfering with monitored applications. Researchers can also add stand-alone loggers via the add logger wizard.

3.3.3 TCP/UDP Loggers

Unlike command-line or stand-alone loggers that record data into log files or databases, external sensors typically communicate their data via a TCP/UDP port. MultiLog also supports logging data through this mechanism. Loggers that use TCP/UDP communication are common in the HCI community and are often incorporated in devices such as eye trackers or EEG systems. MultiLog supports the same add, remove and pause operations on TCP/UDP loggers as with command-line and stand-alone loggers. The primary difference in operation is that TCP/UDP loggers always update in real-time (and do not require a polling interval) - this prevents excess and unnecessary data buffering.

3.3.4 Logger Summary

Various types of loggers are compatible with the MultiLog logging framework including command-line loggers, stand-alone loggers and TCP/UDP loggers as described in the previous sections of this chapter. A subset of these loggers will be used in this thesis' log study which aims to better understand cross-application interaction. Full details of which logging plug-ins are used for the log study is explained in the next chapter.

3.3.5 Set up and Use

MultiLog allows researchers to add any pre-existing research or commercial logging application to its list of active loggers. Loggers are added through a wizard interface by specifying certain information about the logging application such as: its name, log file location or TCP/UDP details if the logger uses sockets, how to start and stop it and an idea of which log lines are required to appear in the output. Once added a logger can be started and stopped via the MultiLog interface. Once all required loggers are set up, deployment mode can be configured ready for deployment onto participants' computers. This allows the researcher to generate a configuration file for the currently active loggers. Once created, this file, along with the MultiLog executable, can be dropped onto multiple participants' machines ready for data collection. The configuration is a plain-text file that can also be manually edited if required.

3.3.6 MultiLog's Parsing of Log Files

MultiLog supports any logging application that produces line-by-line plain text or XML log files or data received on TCP/UDP sockets. MultiLog parses each new log line, extracts the timestamp and treats the remainder of the line as “data”. XML data is flattened into a single line entry (by extracting the elements/attributes marked as “timestamp” and “data” by the researcher at configuration time); consistent formatting for all input streams makes post-collection analysis simpler. TCP/UDP streams are continuously received (and most are appropriately pre-timestamped) and therefore can be added to the MultiLog interface and database in real time. MultiLog only requires that it can read and understand the timestamp; the “data” portion of the log line may be pre-encrypted by the logging application.

MultiLog assumes that log files (or TCP/UDP streams) are continually updated by the logging application and that data is written in a linear manner (i.e. for loggers using a log file the applications do not rewrite or insert lines earlier in the file). The testing (see “Validation”) showed this to be the case for the vast majority of loggers. A re-usable “work-around” to support Microsoft PSR’s unusual output generation was built. PSR’s output file is only generated when the application is stopped, so MultiLog regularly (currently once a minute – determined by the current polling interval) stops and immediately restarts PSR in order to obtain an output file. This feature can be increased or reduced by changing MultiLog’s general polling interval value from the interface.

3.4 MultiLog Validation

To confirm that MultiLog behaved as expected under various conditions three aspects of the system were examined: (1) That a diverse range of existing logging systems work successfully with MultiLog; (2) That MultiLog maintains data integrity; and (3) That resource consumption is acceptable.

3.4.1 Compatibility with Existing Logging Systems

We tested a diverse range of publicly available logging systems to check their compatibility with MultiLog. The selection of loggers, and the results of these tests are shown in Table 1. The majority (23/33) work successfully with MultiLog. Of the remainder that partially worked or did not work the main issues identified were: (1) The log file is locked by the Operating System so is inaccessible by MultiLog (Microsoft Visual Studio (Microsoft Visual Studio, 2014)); (2) The timestamps in the log files measure time since the logger started, not a general measure of time (Inputlog (Leijten et al., 2005), Translog (Hansen, 1997) & WebQuilt (University of Washington)); (3) The logger spreads log data across multiple log files (Mendeley Desktop log files (Mendeley) & Kidlogger (Kidlogger)); and (4) All data in the file is compressed/encrypted so MultiLog cannot access a timestamp to order events (Skype log files (Skype), SoftActivity Keylogger (Soft Activity) & Revealer Keylogger (Logixoft)). The first (locked log file) and last (encrypted file) issues are outside the control of MultiLog. Issue 2 was solved - timestamps from start time - by adding support for MultiLog to use the last modified time on the log file as the timestamp; Issue 3 - multiple log files - will be addressed in future work. Issue 4 is a limitation of MultiLog's approach to extracting log information.

We also included in table 2 output from a text (Google Chrome), XML (Microsoft PSR) and UDP (Arduino light sensor) logger providing an example of the raw output data produced by the logging application and the data after MultiLog has parsed it.

Table 1: A list of loggers tested with MultiLog. Y = logger works as expected, N = logger does not work, ! = logger partially works but there are known issues as detailed in the Notes column.

<i>Logger Name</i>		<i>Output</i>	<i>Notes</i>
Microsoft Problem Steps Recorder (PSR) (Microsoft Corporation)	Y	XML	
Google Chrome in-built logger (Google Chrome, 2014)	Y	Text	
Mozilla Firefox in-built logger (Mozilla Firefox)	Y	Text	
Mozilla Thunderbird in-built logger (Mozilla Thunderbird)	Y	Text	
Drag (developed by MultiLog authors)	Y	Text	Records mouse drags.
Drag-and-Drop (developed by MultiLog authors)	Y	Text	Records Drag-and-Drop actions including the name of the object being dragged, where it was dragged from and where it was dragged to.
Window Switch (developed by MultiLog authors)	Y	Text	Records window switches and the name, dimensions and size of windows.
Clipboard (developed by MultiLog authors)	Y	Text	Records cut, copy and paste actions.
Process Start Monitor (developed by MultiLog authors)	Y	Text	Records the time when a new process was started by the user or system.
WEKA Data Mining (Hall et al., 2009)	Y	Text	
AppMonitor (Alexander et al., 2008)	Y	Text	
Windows Update log files (Windows Update)	Y	Text	
VMWare log files (VMWare)	Y	Text	
Microsoft Outlook log files (Microsoft Outlook, 2014)	Y	Text	

Adobe ARM log files (Adobe)	Y	Text	
Internet Explorer Maintenance (brndlog) log files (Internet Explorer)	Y	Text	
Windows DTC install log files (Microsoft Distributed Transaction Coordinator)	Y	Text	
User Logger (User Logger)	Y	Text	
JEdit (Eklundh & Kollberg, 2003)	Y	Text	
TeamViewer (TeamViewer, 2014)	Y	Text	
Ubuntu log files (alternatives.log, auth.log, dpkg.log, kern.log and syslog) (Ubuntu)	Y	Text	
Tobii Eye Tracker	Y	UDP	Data continuously received on UDP port 11000. Processed by MultiLog in real time (as and when received).
Arduino Sensor	Y	UDP	Data continuously received on UDP port 8888. Processed by MultiLog in real time (as and when received).
Microsoft Visual Studio (Microsoft Visual Studio, 2014)	!	Text	When this logger runs, the log file is locked. As a result, MultiLog can only process it when the logger is stopped and the file is unlocked.
Inputlog (Leijten & Waes, 2005)	!	XML	<p>The timestamp in the logger is a measure of the time since the logger was started not a general measure of time. As a result, MultiLog cannot determine chronological order unless the user/researcher specifies to use the last modified date from the file as the timestamp.</p> <p>Data is created across multiple log files. Only the file listed as the log file when the logger is added will be included initially. In order to include subsequent files, new loggers would have to be added via the Add Logger wizard.</p>
Translog (Hansen, 1997)	!	XML	<p>The timestamp in the logger is a measure of the time since the logger was started not a general measure of time. As a result, MultiLog cannot determine chronological order unless the user/researcher specifies to use the last modified date from the file as the timestamp.</p> <p>Data is created across multiple log files. Only the file listed as the log file when the logger is added will be included initially. In order to include subsequent files, new loggers would have to be added via the Add Logger wizard.</p>

Mendeley Desktop log files (Mendeley)	!	Text	Data is created across multiple log files. Only the file listed as the log file when the logger is added will be included initially. In order to include subsequent files, new loggers would have to be added via the Add Logger wizard.
Kidlogger (Kidlogger)	!	XML/ HTML	Data is shown every minute for a period of one day. After this time, Kidlogger creates a new file and another logger would have to be added to MultiLog through the Add Logger wizard to reflect this.
WebQuilt (University of Washington)	!	Text	The timestamp in the logger is a measure of the time since the logger was started not a general measure of time. As a result, MultiLog cannot determine chronological order unless the user/researcher specifies to use the last modified date from the file as the timestamp.
Skype log files (Skype)	N	Text	Data is hashed/encrypted and MultiLog is unable to un-hash/decrypt it.
PersonalVibe (Microsoft Research)	N	Database	Data is held in a database and MultiLog does not handle databases.
SoftActivity Keylogger (Soft Activity)	N	Text	Data is hashed/encrypted and MultiLog is unable to un-hash/decrypt it.
Revealer Keylogger (Logixoft)	N	Text	Data is hashed/encrypted and MultiLog is unable to un-hash/decrypt it.

Table 2: A list of three loggers (with three different types of output) along with example raw and MultiLog-parsed data.

<i>Logger Name</i>	<i>Raw Data</i>	<i>MultiLog-parsed Data</i>
Google Chrome	[9056:14968:0806/141932:VERBOSE1:resource_loader.cc(335)] OnResponseStarted: https://docs.google.com/offline/backgroundshell#oid=ud3488c2d87270738	08/06/2015 14:19:32.000 OnResponseStarted: https://docs.google.com/offline/backgroundshell#oid=ud3488c2d87270738
Microsoft PSR	<EachAction Time="17:36:43"> and <Description>User left click in "MultiLog.docx - Microsoft Word"</Description>	08/06/2015 17:36:43 User left click in "MultiLog.docx - Microsoft Word"
Arduino Light Sensor (As the logger is written by us, data is constructed in MultiLog's universal format in the pure logger code.)	100	08/06/2015 17:18:34 Data: 100

3.4.2 Data Integrity

Data integrity ensures that MultiLog is recording all the required events and that it is recording all of the data associated with these events. The following method was used for validating data integrity: (1) MultiLog was configured to run one logger without filtering, see Table 3; (2) A series of interactions with the monitored application were performed; (3) The resulting events recorded by MultiLog were extracted from its database and the original log file generated by the logging software was copied; (4) MultiLog's output was transformed back into the format generated by the logging application by an additional piece of software; (5) The original log file and the transformed output from MultiLog were compared using a file difference checker.

We applied this methodology to a variety of loggers as illustrated in Table 3. For plain text output all files were identical except for blank lines (which MultiLog automatically removes) and a handful of characters, such as the single quote, which were removed as they interfere with MultiLog's database.

Nine further trials were performed on XML loggers. This time the raw log file was run through a third-party XML processor to extract the timestamp and data and then formatted to match that produced by MultiLog. The difference checker was then used to check for differences. On observing the output all files were identical as described in Table 3.

Three additional trials were conducted with the Arduino light sensor UDP logger. The functionality was altered to also write logs to the Arduino output window so a comparison between the raw data and the data received in MultiLog could be completed. Data over a one minute period was checked and matched (MultiLog did not drop packets and timestamps were correct).

Table 3: Loggers used for output validation.

<i>Test Logger</i>	<i>Number of test repetitions</i>	<i>Output type (text/XML)</i>	<i>Average length of interaction</i>	<i>Average number of log lines produced</i>	<i>Results</i>
Google Chrome	3	Text	1 minute	7315	All lines identical except those which were omitted automatically by MultiLog because they did not contain a timestamp.
Mozilla Firefox	3	Text	1 minute	883	All lines identical.
Mozilla Thunderbird	3	Text	1 minute	353	All lines identical except those which were omitted automatically by MultiLog because they did not contain a timestamp, those that contained the single quote character which had to be removed to avoid database clashing and non-standard ASCII characters.
Microsoft PSR	3	XML	1 minute	80	Both files identical.
Inputlog	3	XML	1 minute	2192	Both files identical.
Translog	3	XML	1 minute	172	Both files identical.

3.4.3 Resource Consumption

Finally, MultiLog's impact on system performance (% processor time) was tested. Various tests were conducted using an average CPU tool. Table 4 shows average CPU utilisation when running MultiLog with a number of different logging applications. These loggers were run firstly with user interaction with the PC (so events were generated) and secondly with no user interaction (when the PC was idle, but the loggers were still running). All tests were run with a log polling interval of one minute. It was found that, on average, 0.069% CPU was used when running MultiLog with one logger. This increased to 1.390% when running four. It was also observed that increasing logger output directly impacts on performance.

Table 4: Loggers used for performance validation. All tests were run for a period of 15 minutes apart from the Arduino light sensor test which was run for 30 minutes. The time intervals selected were relatively short because each test requires continuous interaction within that application, something current time constraints did not permit. ‘Interaction’ refers to the PC being utilised for the duration of the test. ‘No interaction’ means the PC was idle for the duration of the test. The tests performed with the Arduino light sensor were performed on a different machine due to hardware constraints. The no interaction figure provides a baseline from which the overheads can be calculated.

<i>Configuration</i>	<i>Average CPU usage (%)</i>	
	<i>No interaction</i>	<i>Interaction</i>
MultiLog + 1 logger (Microsoft PSR)	0.0169	0.0698
MultiLog + 2 loggers (Microsoft PSR and Mozilla Firefox)	0.0170	0.6267
MultiLog + 3 loggers (Microsoft PSR, Mozilla Firefox and Google Chrome)	0.1417	0.8925
MultiLog + 4 loggers (Microsoft PSR, Mozilla Firefox, Google Chrome and Mozilla Thunderbird)	0.0525	1.3902
MultiLog + 1 logger (Arduino light sensor) – high frequency data test (every 3 millisecond constant stream)	0.0018	6.8038

3.4.4 Pilot Testing

MultiLog was deployed to a small number of users over a period of ten weeks to observe if it could run over extended periods of time to allow long-term studies to be conducted. MultiLog successfully ran for this period of time collecting and uploading data from anywhere between one and five loggers. This period of deployment allowed us to correct small interaction bugs discovered such as the red “X” wrongly closing the application rather than minimising it to system tray and various start up issues. It was found that over the initial deployment period that web browser logs, for example, were producing a substantial amount of data, producing over 1000 lines in a matter of minutes. Much of this data was simple HTTP requests and not relevant to the research questions this thesis is addressing and so it was decided to remove these for future user studies. When the appropriate loggers had been selected for the more detailed and extended deployment, an average of 2000 log lines per hour was observed with a database size of between 600 to 1000KB when user interaction is frequent. The loggers tested with during initial deployment (with the average log lines per hour in brackets) were: Microsoft PSR (600), Drag-and-Drop (0-10), Window Switch (120), Clipboard (0-20) and Process start (450). Generally, an interaction event is captured by more than one logger (and therefore more than one log line). For example, opening a new window would be captured by the Microsoft PSR (general logging), Window Switch (when the window came into focus) and Process start (when a new process was started) loggers. The same setup and loggers were used for the actual data collection study.

3.4.5 Discussion

MultiLog also provides the ability to log and combine events from outside the native PC environment, for example, from devices such as eye-trackers and EEG sensors. This is supported through TCP/UDP logging (a common input method for external sensors). This further increases MultiLog’s power, allowing researchers to experiment with otherwise complex software and hardware logging system setups. Examples include: investigating how applications on the user’s PC interact with external devices or inputs (from the user) such as eye trackers and finger pressure sensor input. Logs from both the external devices and internal applications can be combined seamlessly in MultiLog to give a rich data set. Researchers could also use MultiLog to understand processing

inefficiencies by combining CPU usage and number of applications open/application interaction or even web server logs and those describing the temperature in a server room for example. MultiLog's flexibility allows non-technical researchers to quickly and easily combine the logging capabilities from pure-software, external sensor-driven and even mobile systems. These features are not used for the data collection related to this thesis but instead were included as an extension to the application for use with other researcher's studies.

A drawback of MultiLog is its inability to capture an insight into the thought process of the users when they were performing certain interactions. MultiLog coupled with additional qualitative data would have enhanced the richness of the data collected as part of the study.

3.4.6 Summary

Event logging in desktop applications provides researchers with a tool to help understand how people interact with interfaces to facilitate improvements for future development. This chapter describes MultiLog, a system which simultaneously controls, gathers, and combines the output from multiple existing research and commercial logging applications. MultiLog does not require technical expertise to configure or deploy. The MultiLog software is available from the website at www.scc.lancs.ac.uk/~woodrufj/MultiLog. The MultiLog software answered the first of the three research questions of this thesis by providing a framework which would allow the data collection aiding the understanding of multi-application use in desktop environments (research question 2).

Chapter 4: Log Study

Following the development of the MultiLog tool, which allowed the collection of relevant data to better understand the use of multiple applications in desktop environments, the software was deployed to participants for the data collection to take place (in the form of a longitudinal log study). This study would use the MultiLog software to collect data on cross-application interaction which will help answer the second research question of this thesis: how and when does multi-application use occur in desktop environments. This chapter explains this longitudinal log study in detail and covers the following subsections:

1. The purpose of the study.
2. The types of data to be collected.
3. The loggers required to “plug in” to MultiLog to collect such data.
4. The length of the study and justification for this length.
5. An overview of the participants and how they were recruited.
6. Start of Study – setup and installation of the MultiLog software.
7. End of Study – removal of the MultiLog software.
8. Qualitative Feedback.

4.1 Purpose of Log Study

Before the longitudinal log study was undertaken using the MultiLog software to analyse how participants used their machines to complete certain tasks, an understanding of which findings were expected to be gained had to be understood from running the log study. The over-arching research goals focus on understanding and improving the ways in which users utilise multiple applications to complete certain

tasks. The study wanted to ensure that it logged as much of this cross-application interaction to enable us to analyse this interaction and to report on the ways in which users employ these applications while using their PCs. During the log study it was hoped to pick up on certain areas of inefficiency on the users or PCs behalf which this thesis would aim to improve upon later in the research. This thesis hoped by collecting a vast array of accurate information from various applications/sources this thesis aims to assist in better understanding user interaction and therefore attempt to provide a more solid basis for improving these interactions. A considerable amount of time was spent writing MultiLog and the associated “plug ins” to ensure that all the data collection abilities required for the study were available.

4.2 Types of Data to be Collected

Before commencing the longitudinal log study the types of data required would need to be understood fully. Before this could happen an abstract understanding about the ways in which users currently utilise their PCs to complete tasks was required. One of the most fundamental- interactions users perform when using PCs are window switches (Tak et al., 2009). These are switches from/to various windows open within their operating systems. As the research is in the area of understanding cross-application interaction the ways in which users switch between these applications is an extremely useful analysis to perform. Consequently, it was decided that the recording of window switches (both user and system-initiated) was required in order to fulfil the requirements for the log study. This analysis furthers knowledge in the area of window switching by providing an up-to-date characterisation of window switching interactions and a comparison to past research.

It was also concluded that as part of the log study data relating to mouse and keyboard interactions should be collected. The ways in which users utilise the mouse and keyboard in various applications could prove interesting in understand the ways in which they use applications together to achieve a task. The mouse and the keyboard are the most common methods of data input in PCs therefore these methods of data input are interesting to the research as they may assist us in understanding how users employ multiple applications to complete tasks. Mouse and keyboard interactions provide ancillary information surrounding interactions on PCs and will pick up various shortcuts

that users utilise, which is extremely helpful when detecting clipboard shortcuts, for example (Ctrl + C, Ctrl + V) which provide additional research in the area of data transfer and PC use in general.

The transfer of data within and between applications was also of interest, particularly data transfers between applications as this may assist in better understanding the ways in which users use multiple applications to complete tasks in a cross-application manner. The two main ways to transfer data between applications are (1) the Windows Clipboard and (2) by way of drag-and-drop. It was decided to log both the Windows Clipboard operations and the drag-and-drop actions the users performed in order to gain a better understanding of their data transfer habits. A characterisation of data transfer usage on desktop computers is not currently available and so this characterisation would be a valuable asset to the data transfer area of related work exploring how users interact with the Windows clipboard and drag-and-drop systems.

File directories accessed by users was also of interest to the research contained within this thesis as the locations in which files are saved could provide valuable information as to the ways in which users are interacting with their computers to complete certain tasks in a cross-application way. For example, if two different files from different applications e.g. Word and Excel are saved in the same directory this could mean these documents are related to the same task. File directories are commonly used items in the desktop environment and so the ways in which they are accessed is relevant to the cross-application area because directories could be task-related and files contained within them could be opened by multiple applications. File directories were logged because of the cross-application nature in which they can be used (multiple applications might store files in the same folder/location).

4.3 Types of Data Not to be Collected

It was decided that various interactions would not be logged as part of the study due to various constraints. Various interaction methods such as in-depth information on the usage of sole applications were not logged as single application use is not cross-application in nature and so not relevant for this research. Interactions such as cross-tab interaction in web browsers were also not logged as these are not cross-application in

nature either, however this is relevant to further research in analysing how users interact across tabs. Other interactions and higher level information such as whom the user e-mailed or which websites were visited were also not logged due to their non-cross-application nature. Other cross-application interactions such as attaching files to e-mails and interactions within virtual machines weren't logged due to restraints in available APIs possibly due to privacy surrounding logging of such actions.

4.4 Loggers Required

Once a full understanding of the types of data required was undertaken, a suitable solution had to be found (or written) to “plug in” loggers to the MultiLog logging framework which enabled data collection of these types of data. From the previous section there were four logging “plug ins” to source/develop in order to provide the required data set for the study:

- Window switching & Windows Explorer file directories.
- Mouse clicks and keyboard entry.
- Windows Clipboard.
- Drag-and-drop.

The following sub-sections of this thesis describe the different loggers in detail.

4.3.1 Window Switching & Windows Explorer File Directories

In order to capture window switches a logging application that could be “plugged in” to MultiLog had to be sought/developed. After much research into the area of window switching loggers it was discovered that a logger which logged window switches and other associated data and could be “plugged in” to MultiLog did not currently exist. Consequently it was decided to develop this logger “plug in” for the study. The window switching “plug in” comes with the MultiLog software and can be enabled and disabled either through the MultiLog interface or through the configuration file if used in deployment mode. An example of the output from the window switching logger is shown below:

Window Switch to: Skype. Process: Skype (500). Window Dimensions: Width: 1382 Height: 744 Position: Left: -8 Right: 1374 Top: -8 Bottom: 736 Status: Maximized

This research developed the window switching logger to trigger a new log line on the detection of a system or user-initiated window switch. When a window switch occurs, the logger captures the window title of the switched to window, the Windows process it is assigned to, the width and height of the window and its position on-screen. The window state (maximised or windowed) is also displayed. This data enables us to know at each window switch where the switched to window is and in what condition it currently is. The window switching logger also reports a list of open windows at switch time so it will record the window switched to and all other open windows at that time. An example of a window open log entry is below:

Window Open: Facebook - Google Chrome. Process: chrome (3744). Window Dimensions: Width: 1382 Height: 744 Position: Left: -8 Right: 1374 Top: -8 Bottom: 736 Status: Maximized Z Order: 3

The window open log entries capture the window title, the associated Windows process, the window dimensions, location and state (maximised or windowed). It also captures the window z order. The z order is the order in which windows are open on the desktop. For example, the window which is on top is always z order 1 and the window immediately behind it is z order 2. Collecting window z order data enables us to know how close to the top (possibly hinting at frequency of use) the given window was.

It was found that the Window Switching logger could easily be extended to record file directories accessed through the Windows Explorer application. Each time a new directory is accessed, a new log line in the output is created. File directories accessed are recorded if they are used in the main Windows Explorer application or in either the “Open” or “Save As” file dialog windows which are opened from other applications (when opening files). An example of an output from file directories accessed is below:

Windows Explorer Path: C:\Users\test\Documents\

4.3.2 Mouse Clicks & Keyboard Entry

The next logging application “plug in” which had to be investigated was one that would record mouse clicks and keyboard input. Various keyloggers existed but the Microsoft Problem Steps Recorder (PSR) was also discovered. The PSR application is designed to record user interactions through the keyboard and mouse in order to be able send this data to Microsoft and/or other technical organisations in an attempt to provide information to assist in resolving a particular issue a user has with their computer. A way that this application could be run from the command-line from within MultiLog was found and it served the data collection requirements well.

A little extra software had to be added to the MultiLog software to start and stop the PSR program and to transform its XML output to add it to the universal MultiLog output database (described fully in Chapter 3). This functionality is included in the MultiLog application and the PSR logging can be started and stopped either through the MultiLog interface or the configuration file if MultiLog is being used in deployment mode. A typical example of output from the PSR logger is as follows:

```
User left click on "New Tab (push button)" in "Google Chrome"
```

The type of interaction (left click, right click, keyboard input) and the type of control clicked along with the window/application name is recorded for every entry. These are dynamically added to the MultiLog database in real time when received by the Microsoft PSR logging application.

4.3.3 Windows Clipboard

The Windows Clipboard is a temporary storage area within Windows which is used for storing text and files, etc., when selected by the user. The three actions the user can perform with the Windows Clipboard are “cut”, “copy” and “paste”. The “cut” action removes the data from the source area and stores it temporarily on the Windows Clipboard. The “copy” action copies the data from the source area and stores it temporarily on the Windows Clipboard. The “paste” action pastes the last “cut” or “copied” item from the Windows Clipboard into the destination area/application. After researching possible logging applications which would record clipboard operations, it was realised a logger which recorded the required information was not already available

and would have to be developed. The clipboard logger records all “cut”, “copy” and “paste” text and file clipboard operations. All recorded clipboard operations are added to the MultiLog database in real time. An example of the output produced by the clipboard logger is as follows:

```
Item: "TEXT" of size: "33" (characters) COPIED to clipboard (via
menus) from: "Google Chrome (chrome (4852))". Window Dimensions:
Width: 1382 Height: 744 Position: Left: -8 Right: 1374 Top: -8
Bottom: 736
```

The type of data (text/file) is recorded along with the number of characters if the item is text. The type of operation (cut/copy/paste) is also recorded along with the source/destination application. The associated window dimensions and location are also recorded for extra clarity.

4.3.4 Drag-and-drop

Drag-and-drop is another form of data transfer in Windows PCs. Drag-and-drop occurs when an item (text or file) is selected with the mouse. The mouse button is then pressed and not released and the item is subsequently dragged by moving the mouse to the destination area. When the user wishes to “drop” the item the mouse button is released and the data is successfully transferred to the destination area from the source area. Drag-and-drop does not allow the user to store the data for later retrieval, instead the move operation happens in real time, moving from the source to the destination window. An example of the drag-and-drop output produced by the “plug in” is below:

```
Item: Test.docx Dragged from: (WINWORD (5263)) to: Input Form
window: Google Chrome (chrome (3744)). The action took:
00:00:01.3455175
```

The data recorded is the item that was copied, the application it was copied from, the item it was copied to and the associated application. The length of the drag-and-drop operation was also recorded.

4.5 Length of Study

The length of the data collection study was an important consideration during the planning stage. The study had to run for a long enough period of time to gain an insightful and accurate dataset from the participants. A period of three months of data collection was deemed suitable for this study. This time period was chosen as it would not only give an indication of user interactions over an extended period of time but would also prove useful to determine whether user interaction trends changed over time. Due to time restrictions on the MPhil a period of over 3 months was not chosen as selecting a longer period of time than this would not give enough time for other relevant research to take place. Consequently, the study was undertaken for a period of 3 months from September 2015.

4.6 Participant Overview

Participants were recruited from family, friends and through connections within the School of Computing & Communications at Lancaster University. Various e-mails requesting assistance with the study were sent out to the department and once participants registered their interest in the study a participation information pack was sent which explained the study in detail and what they needed to do. 17 computer users from a variety of different backgrounds (11 frequent - more than 50 interaction days - and 5 female) took part in the user study for a period of 90 days. The age range of participants was between 18-74 years. Seven users had single screens, eight had two screens and two had three screens. Participants used a variety of Microsoft Windows operating systems with eight using Windows 7, three using Windows 8.1 and three using Windows 10. Seven users had laptops and eleven desktops. The table below shows an overview of all participants, the type of computer they used and a brief overview of the interaction they performed. An interaction day/hour is a standard day/hour where the user has performed some interaction on their computer. An interaction session is defined as a period of consecutive hours interaction (measured by detecting window switches). A session is over when an hour with no interaction is encountered following one or more hours of consecutive interaction.

Table 5: Log study participant overview.

Participant No.	Days Running	Days Interaction*	Hours Interaction (hrs)	Interaction Sessions (#)	No. of Screens	Operating System	Computer Type
1	90	89 (98%)	1321	333	1	Windows 7	Laptop
2	90	80 (88%)	591	145	3	Windows 10	Desktop
3	90	72 (80%)	435	122	1	Windows 7	Laptop
4	90	70 (77%)	249	111	1	Windows 10	Laptop
5	90	64 (71%)	594	252	2	Windows 8.1	Desktop
6	90	63 (70%)	563	88	3	Windows 7	Laptop
7	90	60 (66%)	361	104	2	Windows 7	Desktop
8	90	58 (64%)	404	144	1	Windows 7	Desktop
9	90	58 (64%)	279	151	1	Windows 7	Desktop
10	90	54 (60%)	189	78	1	Windows 8.1	Laptop
11	90	52 (57%)	330	62	2	Windows 8.1	Desktop
12	90	43 (47%)	225	69	1	Windows 7	Desktop
13	90	42 (46%)	228	76	2	Windows 7	Desktop
14	90	30 (43%)	75	48	2	Windows 10	Desktop
15	90	34 (37%)	178	51	2	Windows 7	Laptop
16	90	31 (34%)	132	48	2	Windows 10	Desktop
17	90	27 (30%)	139	52	1	Windows 7	Laptop
Mean:		55 (61%)	370	113			

* (% of days running in brackets)

4.7 Start of Study

Prior to the study commencing a participant information form was sent out and the study fully explained to each participant (in person in most cases). All information given out to participants in document form was approved by the University Ethics Committee and the MPhil supervisor. Once the documentation had been completed and the permission forms signed the installation of the software could begin. For those participants who felt confident enough to install the software themselves, the installer file and a set of installation and operating instructions was sent to assist them in the software installation. For those who were not confident to conduct the installation themselves a house/office visit was arranged for the researcher to install the software on their behalf. The installation procedure was step-by-step and was described in detail in the supplied

instructions. There were no reported issues with installing the software. Once the software was successfully installed MultiLog immediately started collecting the relevant data due to its pre-configurable deployment mode.

4.8 End of Study

Once a period of three months had elapsed with each participant they were informed that the study had finished. Those who were confident enough to remove the software themselves were again sent instructions with details as to how to do this. For those participants who were not confident a home/office visit was again conducted in order to remove the software from their machines.

4.9 Qualitative Feedback

On conclusion of the study qualitative feedback in the form of a demographics survey was conducted. Participants were invited to complete an online survey (or paper alternative) to provide information surrounding their gender, age, employment status, level of education reached, job title/subject areas and other information relating to their computer type (desktop/laptop), operating system (Windows 7/8/8.1/10), how they interacted with their computer (trackpad/mouse), the number of screens they used and the types of applications/tasks they performed on their machines. A brief overview of the results of this demographics survey is contained within section 4.5 of this thesis.

Chapter 5: Understanding User Interactions

After gathering a large data set of 17 participants' interactions with their PCs over a ninety-day period the next stage of the research was to undertake an analysis of this data. An analysis on window switching, data transfer and file management was conducted. This thesis wanted to understand how users currently interact with their machines as this would pave the way for understanding how multiple applications are used to complete tasks on participants PCs. Understanding the data transfer between these applications is also a very important tool in the overall understanding of cross-application interaction on PCs. Another set of analysis was performed on data transfer data, in particular clipboard and drag-and-drop data, to better understand the types of data that are transferred between applications on PCs. Finally, further analysis into file directories accessed was undertaken to provide an analysis of how file directories are currently accessed and used. The ways in which participants used files and the locations they were saved could also provide important information surrounding cross-application interaction. This chapter presents results in x sections:

1. Window Switching.
2. Data Transfer.
3. File Directory Access.

5.1 Window Switching

Window switching is a frequent task undertaken by users of all desktop operating systems in order to access software and resources (Tak et al., 2009). Past work (Tak et al. (2009), Gaylin (1986), Tak (2011) and Hutchings et al. (2004)) has explored various aspects of window switching. Gaylin (1986) focussed on creating an “empirically based

windowing benchmark” and observed users for approximately 20 minutes, finding participants used “cycling through windows” (a method similar to Alt + Tab) most frequently when switching between windows. Tak et al. (2009) reported high window re-visitation rates and methods of switching windows (direct click and taskbar selection were the most frequent), window depth level, spatial, and temporal data. Hutchings et al. (2004) focussed on window visibility and methods of switching. They found the average amount of time a window was active was 20.9 seconds and hypothesised that multi-screen users accessed windows through a direct click more often, and through the taskbar less often, than single screen users. Users’ desktop environments have continued to evolve since these studies. Most significantly, users have an increasing number of screens with an increasing screen area (Czerwinski et al., 2003).

To increase the community’s understanding of the role of window switching in everyday desktop computing this thesis set presents a re-visitation and expansion of past results to characterise current behaviour. Such a characterisation helps researchers to identify areas of inefficiency and aid in the development of new tools. To do this 17 participants’ user interactions were logged with their windowing systems over a period of 90 days using MultiLog (Chapter 3). This chapter then analysed three categories of interactions: (1) window switching methods; (2) location and size of windows and; (3) number and state of windows. Where the data exists the current observations are compared to those detailed in previous work (Tak et al., 2009).

The key observations from the characterisation are that: window switching methods have remained constant over time; clicking in the destination window remains the most frequent switching method; placement of windows on-screen has remained constant for both single and multi-monitor users; users do not search deep in the window stack to find a required window indicating that they interact with a small group of windows extensively. This thesis provides characterisations of: the on-screen placement of different types of windows, maximised vs. non-maximised windows, application-specific switching rates and the number of applications and windows open on single and multi-screen computers. These insights should aid future interface designers in the development of tools and techniques for window and application switching.

The remainder of this section will explore related work in the area, outline out methodology, and present an empirical characterisation of window switching activity

in three categories: (1) window switching methods; (2) location and size of windows and; (3) number and state of windows. The implications of these findings are then discussed.

5.1.1 Results

5.1.1.1 Window Switching Methods

Window Types. 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 process¹. To begin window switches based on window type are classified. This allows us to understand users' switch behaviour between different components of their applications. It was found that, averaged across participants, 55% of windows switched were to main (top-level) windows and 41% were to sub-windows (options, configuration) as detailed in Figure 3. User-directed messages (dialog windows and progress windows) are switched to the least (both 1%).

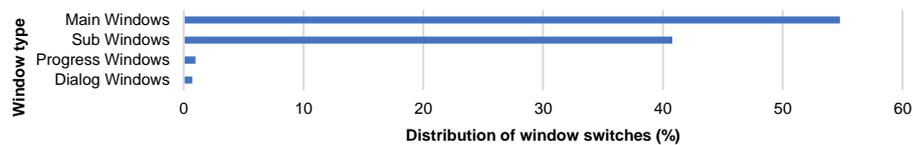


Figure 3 Window switches by type (average across participants).

Gaylin (1986) categorised windows into the following categories: general purpose, informational, clock, and inactive. For comparison to this thesis' dataset general purpose windows are mapped to the main- and sub-window categories and informational windows are mapped to this chapter's progress and dialog windows. For comparison the results from (Gaylin, 1986) and this chapter's characterisation are shown in Figure 4. Current window type usage continues to follow the trends identified

¹ Switches that were identical in timestamp, window name and size, those that were switches to a minimised window and those that were duplicate switches, were removed from the data set before analysis commenced. This can occur when participants accidentally miss the target window or if the switch was reported in duplicate by the operating system. Switches to minimised windows can occur due to the Windows APIs incorrectly reporting a switch when a user minimises a given window.

by (Gaylin, 1986) with switches to main and option windows accounting for the majority of interactions.

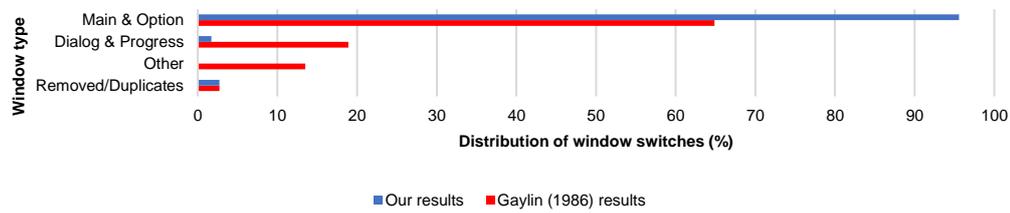


Figure 4: Window switches by type (average across participants) using the Gaylin (1986) categorisation.

Window Switching Methods. For each window switch the method the participant used to bring it into focus was logged. Eight methods of switching were recorded (see Table 2) and matched the 230,439 total recorded switches to these categories². Figure 5 shows a breakdown of these switches by initiation method. Those switches that could not be categorised into one of the ten categories were placed in the “Other” category (0.5% of switches on average across all users).

Table 6: Types of window switch.

Method	Explanation	Type
ALT + TAB	Holding the ALT key down and consecutively pressing the TAB key until the desired window is selected.	Intentional
Dialogs	A click on a dialog window.	Intentional
General Window Click	A click inside the perimeter of the destination window or a non-button click inside a window that triggers the opening of a new window (e.g. double clicking a file).	Intentional/Indirect
Taskbar	A click (or a series of clicks if the application is stacked) on the Windows Taskbar ³ icon of the desired application.	Intentional
Application Button	A click on a button within an application that is not “minimise” or “close”.	Indirect
Close Button	A click on the window’s close button (located on the top right bank of buttons).	Indirect

² 1.9% of switches were removed from this analysis due to the ambiguity or erroneous reasons stated earlier.

³ <http://windows.microsoft.com/en-gb/windows/taskbar-overview>

Minimise Button	A click on the window's minimise button (located on the top right bank of buttons).	Indirect
Other	An action performed by a user that could not be grouped into the above categories.	

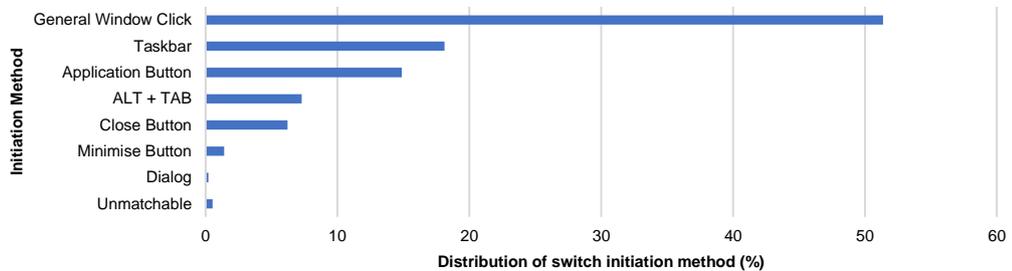


Figure 5: Window switch initiation method (average across participants).

The most common switching methods were general window click and taskbar followed by application button and Alt + Tab. The most common switching actions were visual direct interactions from the user. For example, the user clicking on the application in the taskbar or clicking directly inside a window that is already partially visible. Non-visual indirect interactions (such as Alt + Tab) were less regularly used among participants.

Tak's previous study (Tak et al., 2009) collected similar data on window switching initiation. For comparison Figure 6 compares Tak et al. (2009) and this study's switching initiation methods broken down by single- and multiple-display setups.

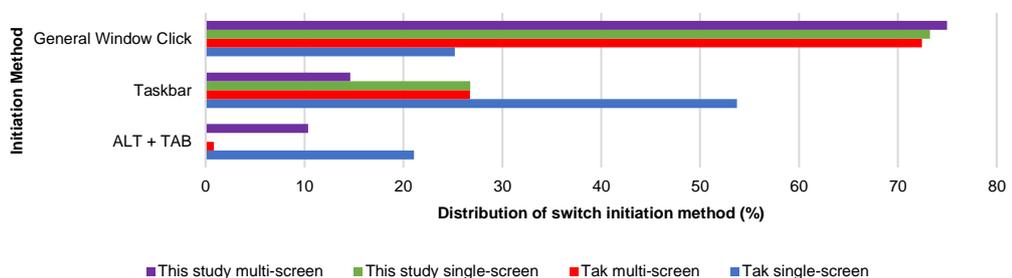


Figure 6: Window switch initiation method by display setup using the Tak et al. (2009) classification.

Similar trends of use exist in both datasets. For multi-screen users general window click was more regularly used in both instances with taskbar the second most frequent and Alt + Tab the least common. For single-screen users trends differ. The data from this thesis' log study shows the most common switching method was general window click with Alt + Tab the least utilised whereas Tak et al. (2009) reported the taskbar as the most common and Alt + Tab least used method. The main difference between the two data sets is the increase in switches using the general window click method and the decrease in taskbar-initiated switches, for both single- and multi-screen users.

Hutchings et al. (2004) found that users of both single- and multi-screen setups use direct window click more than taskbar, a result that is also validated here.

Window Switching Frequency. A timestamp of every window switch was logged as part of the study allowing the analysis of the frequency with which users switched between windows. To do this all switches for each user were extracted and the time between each switch was calculated. Frequency categories with one-minute buckets were created and each switch assigned to the relevant category. If the time between two switches was more than 30 minutes it was assumed this was a case when the user spent time away from their computer (during the night for example) and subsequently excluded this time-chunk as it does not provide a useful insight into switching behaviour.

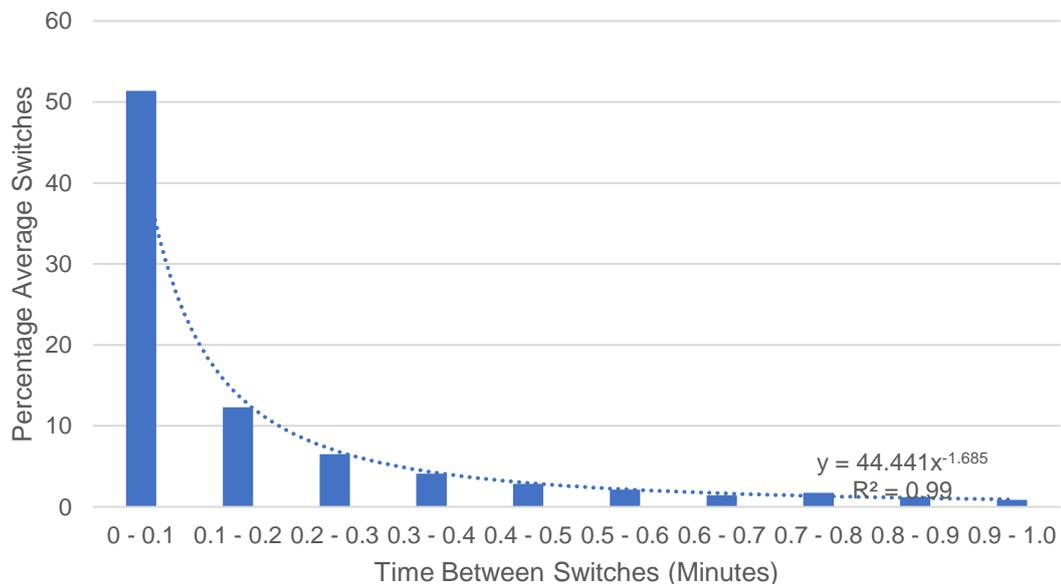


Figure 7: Average time between window switches (minutes).

Hutchings et al. (2004) presented results showing the average time a window was active was 20.9 seconds. This study's data shows an increase of 33 seconds to 54 on average showing users are spending more time interacting with windows before moving onto a different one.

5.1.1.2 Position & Size of Windows

This section analyses user interaction relating to the position and size of open windows

Position of Windows. The analysis of window position is broken down by window category (main, sub, dialog, progress) to reflect their different use and on-screen location. Heat maps were generated for each user for each window type that show a count of the number of windows in a certain position. The reddest areas are where the highest number of windows were located. The reddest areas on the heat maps equate to 100% of windows being positioned there whilst the bluest areas account for 0% of windows. For main and sub windows the heat map percentages are calculated across all windows, whereas for dialog and progress windows, they are calculated on total dialog/progress windows. This is because of the low number of dialog and progress windows seen in the study which caused a blue washout. For multi-screen users the same scale for all screens was used using the highest number of windows across all screens as the brightest red colour. Patterns of window placement were then identified and reported in the form of heatmaps for illustration for each participant.

Main Windows. Three distinct patterns were found for main window positioning: (1) *Maximised:* windows predominantly filled all available screen space (except the bottom task bar), four users; (2) *Windowed:* windows predominantly only partially-filled the screen, ten users; or (3) *Off-centre:* windows were either maximised or, more regularly, appeared to one side of the screen, three users.

Those users who used mostly Windowed mode for windows were those who had more than one screen, while those using a mostly Maximised configuration were those with only one screen. There was no clear distinction between users of laptops and desktops and the type of window configuration used.

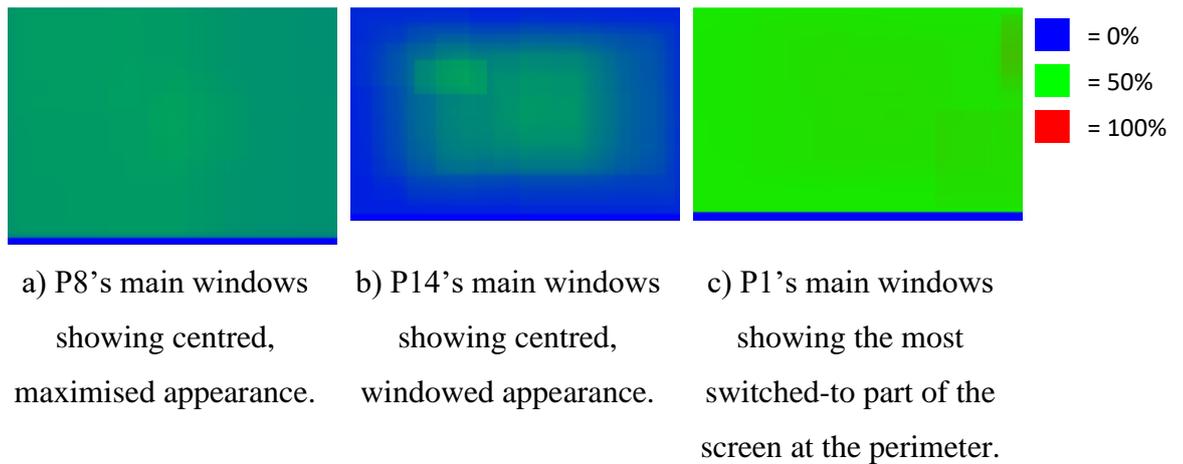


Figure 8: Heat maps for three users' main window distribution.

Sub Windows. Sub window positions were aligned to one of two categories: (1) *Maximised*, Figure 9a and; (2) *Windowed*: ten users, Figure 9b. Those that were windowed appeared mainly in the centre of the screen expanding outwards. Those configurations that did not fit into the above categories consisted of clustered perimeter configuration and off-centred appearance (Figure 9c). Participants who primarily used maximised sub windows generally had a cluster of smaller sub windows appearing in the centre of the screen too. It was also found the number of screens did not have an effect on the choice of windowed or full-screen mode.

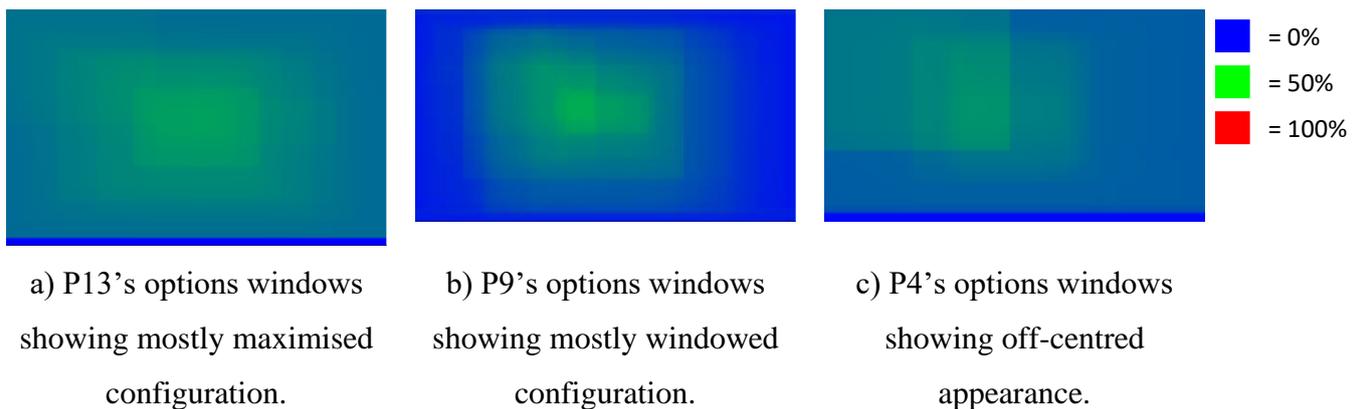


Figure 9: Heat maps for three users' sub window distribution.

Those users found to be using mostly maximised windows were not the same group of users who used mainly maximised main windows, possibly hinting that the size of options windows was set by the application and was possibly not configurable.

Dialog Windows. The majority of dialog windows were small (average size: 200 x 200 to 300 x 300 pixels squared) and appeared near the centre of the screen. Participants with multiple screens had dialog windows near the centre of both screens – Figure 10a shows the primary screen of a multi-screen user’s setup.

In four cases the dialog windows appear in other locations such as in the corners of the screen or along the perimeter (Figure 10b). In two cases these windows appeared elsewhere on the screen (Figure 10c). This behaviour of window placement could be explained by the location of the application that generated the pop-up. In most cases they appear in the centre (the default location). It was found within single-screen users that dialog windows are more likely to appear in the centre or near-centre of the screen (five users). Multi-monitor users had a wider distribution of dialog window positions.

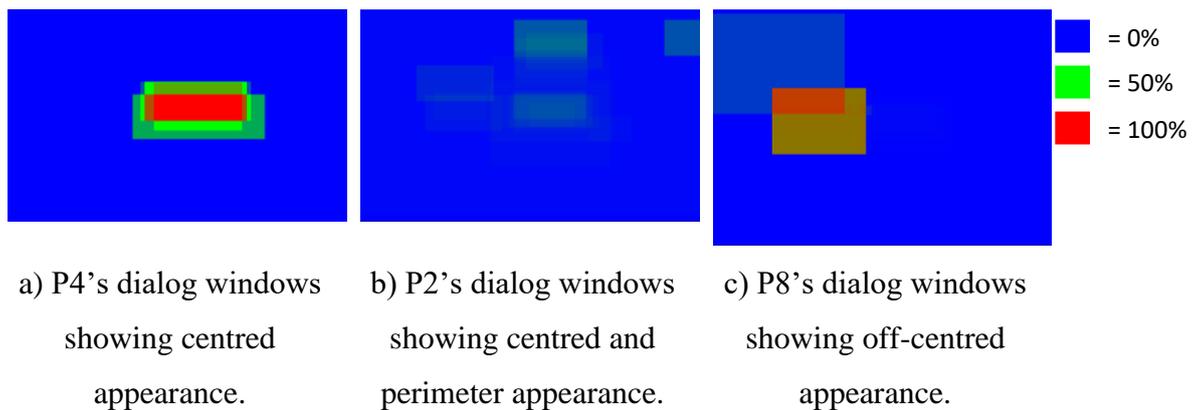


Figure 10: Heat maps for three users' dialog window distribution.

Progress Windows. Progress windows were small (on average 300 x 300 to 400 x 400 pixels squared) appearance followed three main patterns: (1) Centered, six users, Figure 11a; (2) Predominately centered, four users, Figure 11b or; (3) Predominately off-centred, three users, Figure 11c.

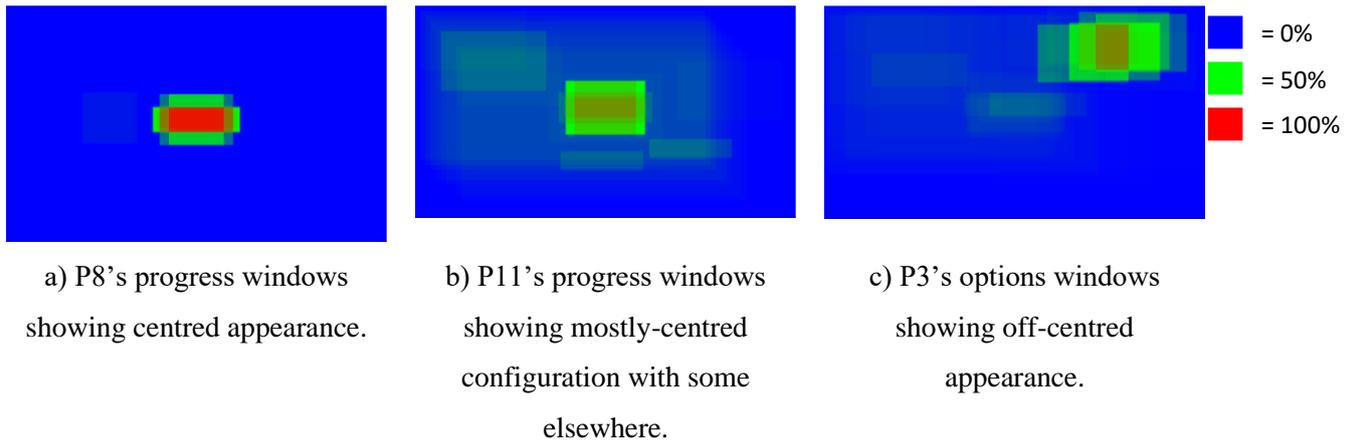


Figure 11: Heat maps for three users' progress window distribution.

Comparison to Previous Characteristics. Tak (2011) also produced heat maps that conveyed the most used areas of the screen (reproduced in Figure 12). Tak did not sub-categorise all windows but similar trends exist between both sets of heat maps. The most frequently used part of the screen in both cases was the centre with occasional peripheral activity. Similarities between Tak's results and these results were found as some users interact with both screens frequently and there are some instances where little interaction on the second screen was observed. Results from this study are shown in Figure 13.

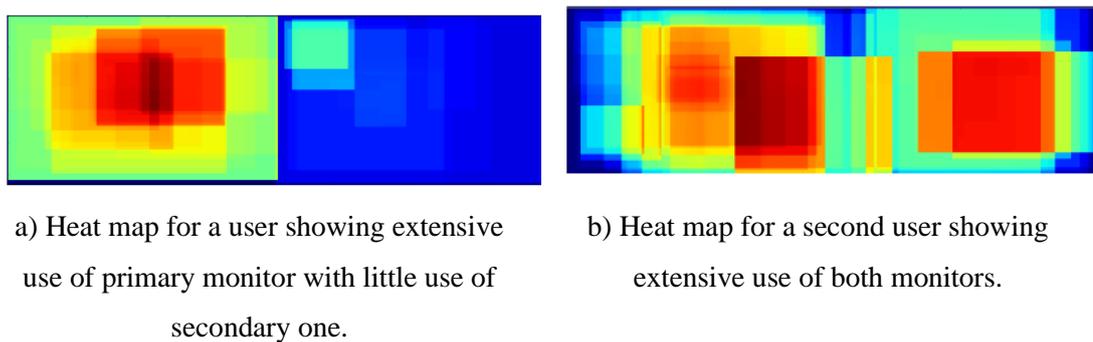
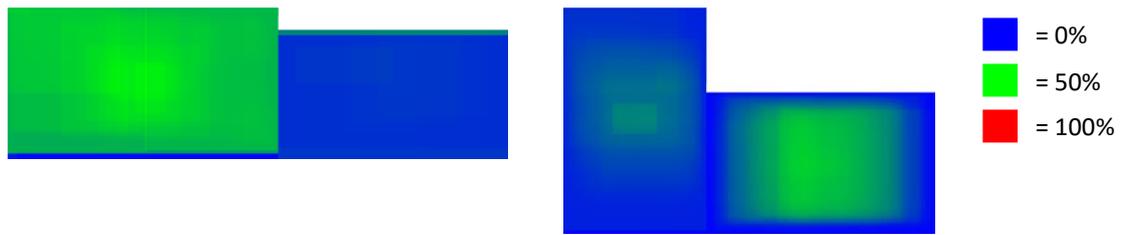


Figure 12: Heat maps from Tak (2011) results.



a) Heat map for a user showing extensive use of primary monitor with little use of secondary one.

b) Heat map for a second user showing use of both monitors.

Figure 13: Heat maps from multi-screen users.

Use of Multiple Screens. For participants using multiple screens their window switching activity across their displays was analysed. To do this it was determined whether the destination window was on the same or a different screen to the source window. 82% of window switches occur to a destination window on the same screen. This shows a tendency for users to focus their activities on a single screen rather than interleaving activities across both screens. Figure 14 shows the full results. Windows that could not be categorised or spanned more than two screens were placed in the other category.

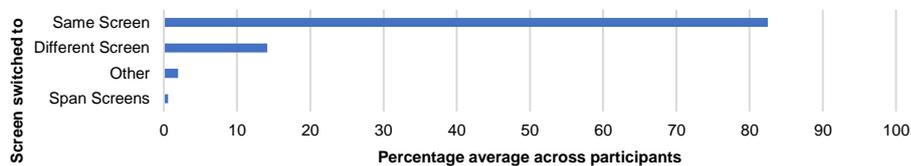


Figure 14: Window switches to the same or different screens for multi-screen users.

Window Movement & Re-sizing. When each window was switched to it was logged whether the window was in the same location as it was when previously used (84%) or whether the location since the last switch had changed (16%). Tak (2011) collected data on the frequency of window movement and found on average 8 moves for single-screen and 15 moves for multi-screen users per day. Both sets of results show a low frequency of location changes for windows, following the same trends as identified in this thesis.

Data relating to the frequency of window re-sizes was also collected. For each unique window the switches to it were parsed, calculating whether there was a size difference. For example, if given window A was re-sized at 12:00:01 and then again at 12:05:01,

the time between re-sizes would be five minutes. One participant re-sized windows frequently (every 96 seconds) where as other participants rarely re-sized windows (every 5.2 minutes). Re-sizes that were higher than 30 minutes apart were excluded to account for participants who left their machines switched on during periods of inactivity. Tak (2011) produced data on window re-sizes shows around 4 re-sizes for single-screen and 7 re-sizes for multi-screen users per day. This shows very infrequent re-sizing compared with the data which shows on average 3.6 minutes between re-sizes (more frequent).

Window State: Maximised or Windowed. On-screen windows can be in one of two states: maximised (covering the entire active screen) or windowed (covering a subset of the active screen). At switch time the window state of the destination window was logged. The majority of windows are used in windowed mode (Figure 15) with single-screen users more often using maximised windows.

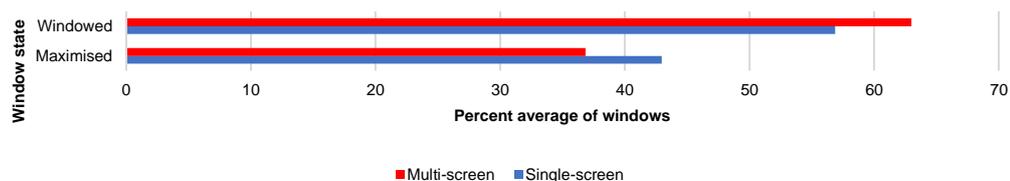


Figure 15: Percentage average of windows in states (by single- and multi-screen participants).

5.1.1.3 Number & State of Windows/Applications at Switch Time

Window Stack Depth. The window z-order⁴ describes how windows on the desktop are stacked from foreground to background. Each window is numbered in increasing order from the foreground window backward. An example is shown in Figure 16. For each window switch the depth level of the window was recorded with an integer value.

⁴ <https://msdn.microsoft.com/en-us/library/windows/desktop/ms632599%28v=vs.85%29.aspx#zorder>

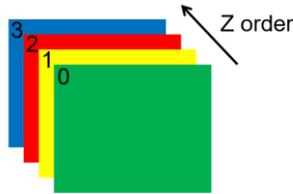


Figure 16: Example of Windows z order.

The results (Figure 17) suggest participants most-regularly switch only a few levels deep in the z order when selecting a new window. 86% of switches are to levels 1, 2 or 3. This suggests participants work frequently on a small set of windows which are normally at the top of the window stack. Z order 2 is the window underneath the top-most window. Z order 1 is the window underneath the top-most window. Z orders of 0 were recorded as participants switched to windows using the “always on top” feature such as Window Task Manager. These applications are forced by the operating system to be at Z order 0 continuously. The highest z order recorded was 41.

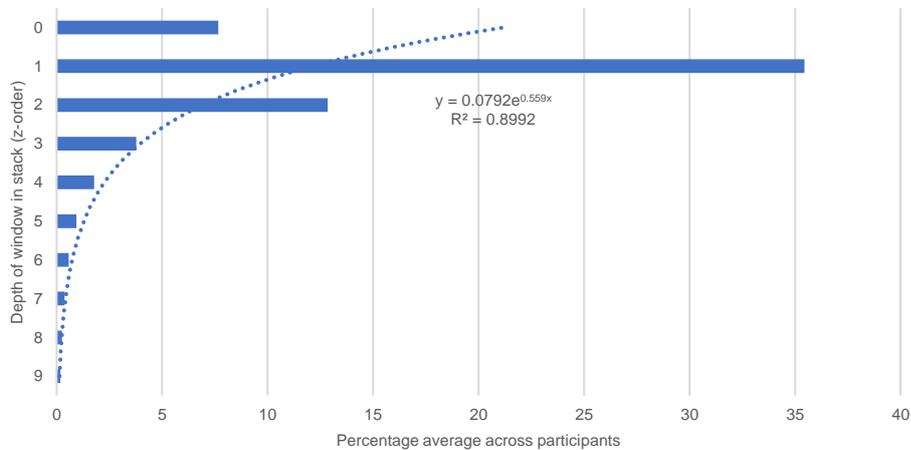


Figure 17: Depth of target window in window stack.

This data was compared to past findings by Tak (2011) who also logged and presented the z ordering of windows. Tak’s results are shown in Figure 18.

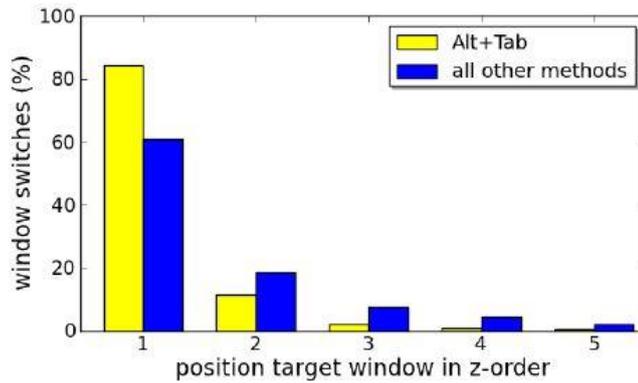


Figure 18: Position of the target window in the z-ordering of windows (Tak).

Ignoring windows switched to using the “Always on top” feature users in the study (Figure 16) followed the same trend of z-order accesses as Tak’s work (Figure 17). Windows at the top of the stack (most recently used) continue to be the most regularly accessed.

Window Re-use. Window re-use is common: 82% of all windows are returned to during their lifetime (rather than used-once and closed). It was observed some applications exhibited a high rate of switching (Figure 19) especially email clients and specialist applications. Tak (2011) collected data relating to unique windows, but by days, not across the whole study. They found 55% of windows (excluding dialog boxes and pop-up windows) are unique. This thesis’ study observed a much smaller uniqueness value (18%) indicating an increase in window re-use since Tak’s observations.

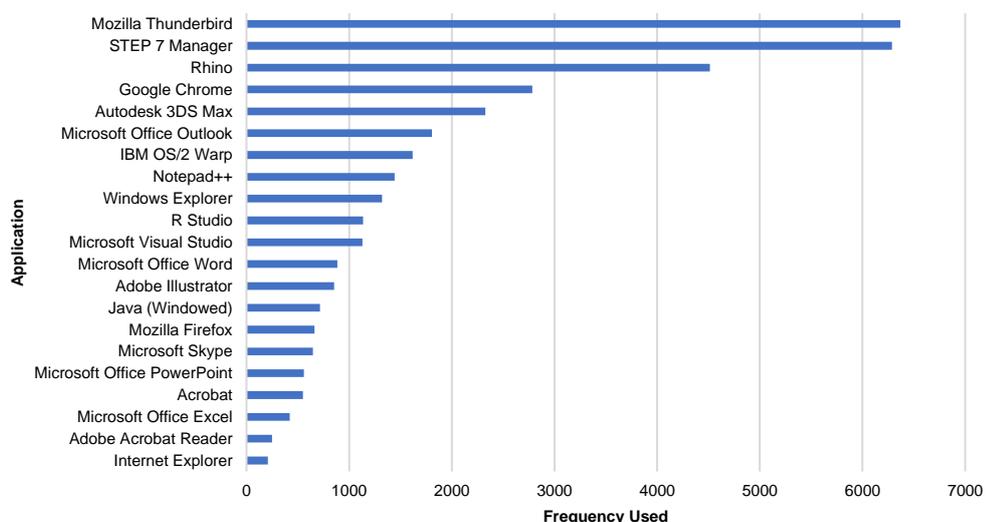


Figure 19: Sum total (across participants) of applications switched to.

Number of windows and applications open. At window switch time the number of windows and applications that were open (an application can have more than one window open) was recorded split by single and multi-screen users. The average for each single and multi-screen user was calculated and then averaged. The most frequent number of windows open at switch time was 7 for single-screen users and 11 for multi-screen users. The most frequent number of applications open at switch time was 6 for single-screen users and 8 for multi-screen users. Figure 20 details these findings more closely. From both figures it is clear that the multi-screen users have more applications and windows open: further investigation is necessary to understand if this is due to increased screen real-estate or the increased likelihood of multi-screen users being power-users. Those utilising a computer more regularly (those who worked with them all day and students/researchers in the Computer Science field) tended to be multi-screen users whilst the occasional home users tended to only use one screen.

Tak (2011) also collected data relating to the number of windows open. Their results are shown in Figure 21. Single-screen users had an average of 6 windows open while multi-screen users had an average of 11. Both sets of results closely align and the new results continue to follow the trends set by Tak (2011). Hutchings et al. (2004) shows that users had eight or more windows open 78.1% of the time—indicating time has had little effect on the number of windows open despite advances in screen size and real-estate.

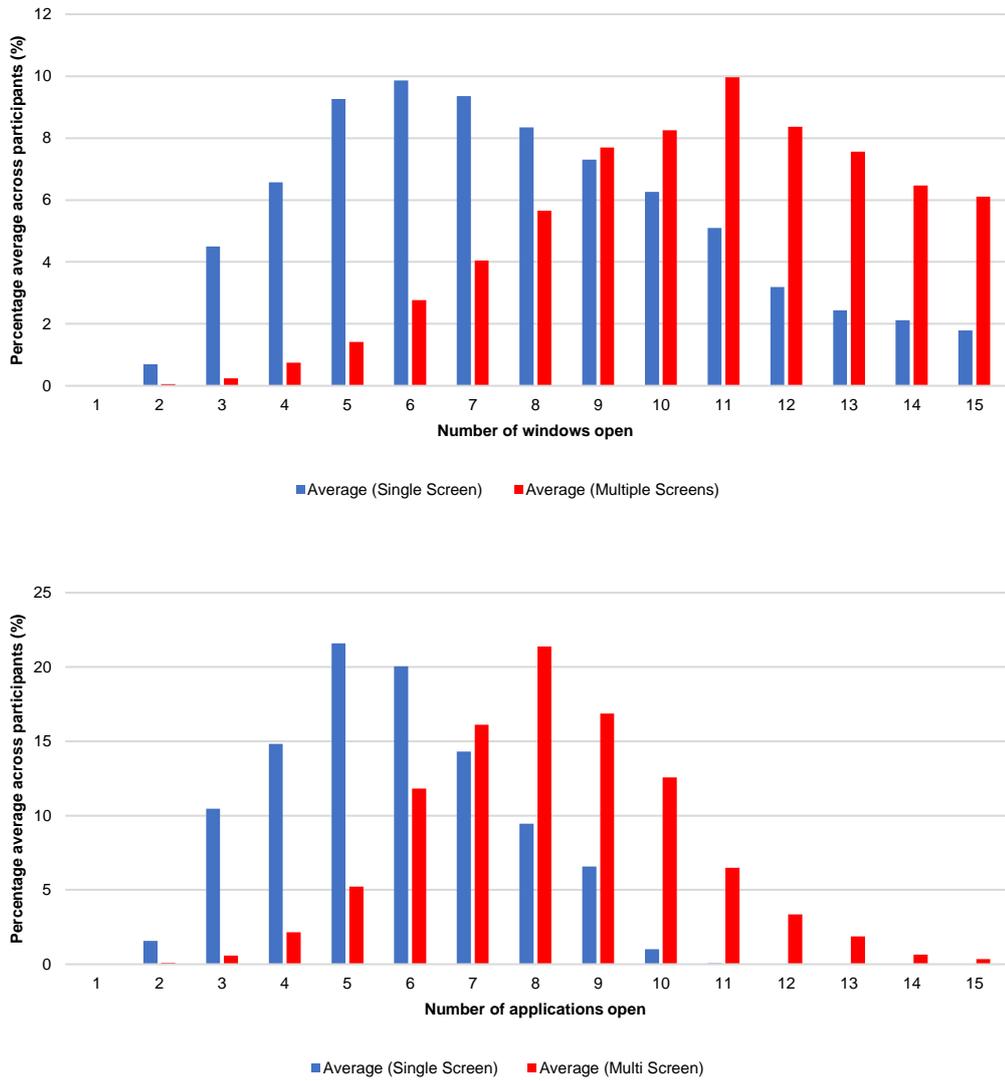


Figure 20: Distribution of windows (top) and applications (bottom) open at switch time.

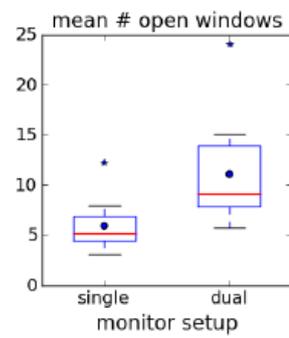


Figure 21: Tak (2011) results showing average number of open windows.

5.2 Data Transfer

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 lack of understanding means human-computer interaction researchers cannot judge the efficiency and effectiveness of these commands or understand areas where efficiencies can be further developed. People who regularly use multiple applications concurrently would also benefit from empirical evidence to assist in the development of cross-application, task-centric interaction tools, as data transfer interactions create an implicit link between the source and destination applications.

Past work (Stolee et al., 2009) explored end-user clipboard interactions, focusing on application and usage patterns. Chapuis and Roussel (2007) and Faure, Chapuis, and Roussel (2009) focused on developing new techniques to aid users when data transfer operations become complicated. Miryung et al. (2004) conducted a field study to explore how programmers utilise clipboard operations while developing software. Drop (2007) and Sunderaraman (2013) focus on the next generation of drag-and-drop actions – those undertaken on the web, while other research (Shneiderman and Kang (2000) and Jia et al. (2006)) provides tools to aid user efficiency when completing repetitive tasks such as labelling photographs. The majority of previous research has focused on developing new data transfer techniques in the case of Stolee et al. (2009).

To better guide interaction technique development this chapter aims to understand the use of the clipboard and drag-and-drop to transfer data within and between applications in an unmodified desktop computer environment. The resulting characterisation is broken into four main areas: (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.

Overall, 31,567 cut, copy and paste actions and 3,432 drag-and-drop actions were collected across all of the participants. 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 and in line with previous work; (3) Text was the most frequently transferred item on the clipboard, while file transfers were less frequent; (4) Keyboard initiation is more frequent than mouse initiation for clipboard operations.

5.4.1 Data Analysis & Results

Following the 90-day study period (fully described in Chapter 4) the logs were collated and prepared for the analysis stage. The analysis took two stages: (1) filtering superfluous log lines and merging individual log files into one “master” log file per user and (2) scripted processing of individual users’ data sets. For timing purposes an interaction hour is defined as any hour in which any user interaction on the user’s computer was detected. Drag-and-drop operations whose duration exceeded one minute were removed from the data set before analysis began.

5.2.2 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.

5.2.2.1 Frequency of Data Transfer

Clipboard Operations

The total number of clipboard (copy, cut and paste) operations over the study period was 31,567. Of the total number of operations, 12,677 (40.1%) were copies, 2,680 (8.4%) cuts and 16,210 (51.3%) pastes. Figure 22 shows this interaction breakdown as an average across all users. These figures show, in total, an almost equal split in data transfer to and from the clipboard.

On average, participants used the clipboard 5.3 times per hour. Eight participants used the clipboard infrequently (on average less than 5 operations per interaction hour), five participants used the clipboard moderately (on average between 5 and 10 times per interaction hour) and four participants utilised clipboard operations on average over 10 times an hour.

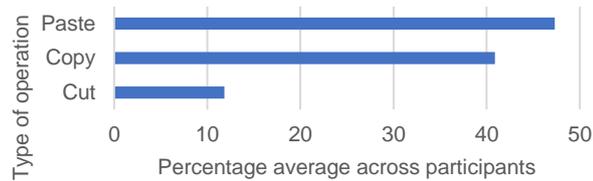


Figure 22: Breakdown of clipboard operations by type.

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 before it can be used. The number of unused cut and copy operations was also recorded. The average number (across participants) of unused operations was 1.6 per interaction hour. This thesis' 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 et al. (2009) reports 19% of their cut/copy actions are unused. Although this study observed a slightly higher rate of “unuse”, this behaviour has not changed significantly.

cantly.

Drag-and-Drop Operations.

A total of 3,432 drag-and-drop operations were recorded over the study period. However, unlike clipboard operations where users fell into four bands of usage it was found that many users either did not use drag-and-drop (1 participant) or used it infrequently – less than 1 operation per interaction hour (14 participants) and two users used drag-and-drop moderately (on average over 3 operations per interaction hour). Figure 23 details this distribution of use, which adheres well ($R^2=0.91$) to an exponential decay model.

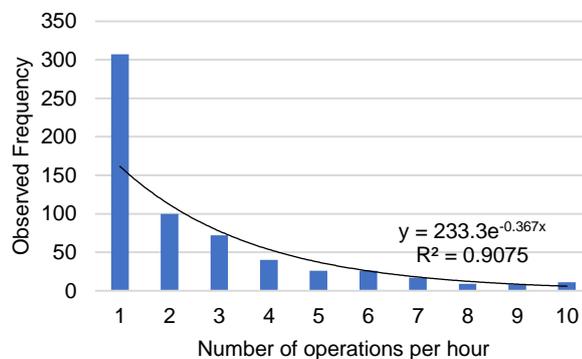


Figure 23: Distribution of the number of drag-and-drop operations performed per hour across all participants.

Summary

Overall, it was found that, on average, the clipboard is utilised more frequently (2 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. It was found that “paste” was used more often than “cut” or “copy”. The use of drag-and-drop was more variable than the clipboard with some participants (P1) not using it at all and others (P17) using it frequently. In their study of programmers using cut-and-paste, Miryung et al. (2004) report 16 instances per hour of use; in comparison, this study saw only 5.3 actions per hour. This difference is due to the study setups. Miryung et al. (2004) studied people for a short duration in their work environment performing particular tasks; this study captures a wide range of both work and leisure use.

5.2.2.2 On-Screen Location & Source/Destination Applications

For each clipboard and drag-and-drop operation the on-screen location and the applications in which the operation occurred was also recorded. The copy/cut and associated paste operations were then paired together for this section.

Screen Location: Clipboard

For multi-screen users it was found that 87% of all copy/cut and paste pairs occurred on the same screen with 12% of actions occurring across different screens. 1% of operations were to or from windows that spanned multiple screens.

Screen Location: Drag-and-Drop

For multi-screen users it was found that 98.69% of all drag-and-drop operations occurred on the same screen and only 0.37% of actions had a 'drop' location on a different screen to the initial drag. A further 0.92% of actions were on windows that spanned multiple screens.

Source/Destination Application: Clipboard

For each clipboard operation the source and destination windows and applications were also logged. For clipboard operations 70% occurred within the same application (Figure 24) and for Drag-and-Drop operations 97% occurred within the same application.

To understand in which types of applications this data-transfer occurred applications were grouped based on their primary function (Figure 25). Previous work observed word processors and web-browsers as the most frequent employers of the clipboard (Stolee et al., 2009). Results confirm these previous findings, with web-browsers and word-processors both occurring in the top six most frequently used for these operations (Figure 25).

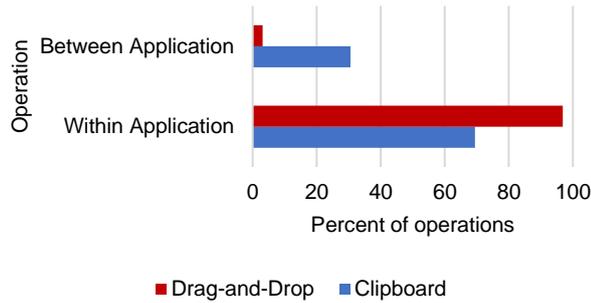
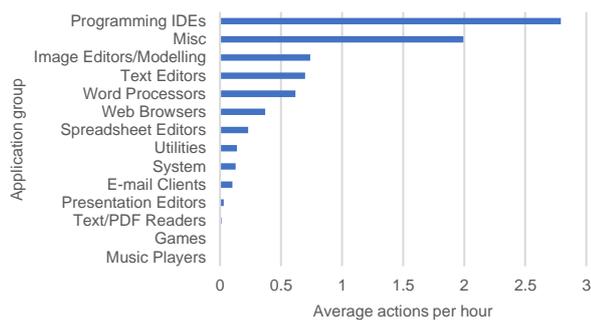


Figure 24: Comparison between within and between application data-transfer operations (average across participants).

Drag-and-Drop

Drag-and-drop results consisted of operations from a small number of users within which a diverse range of operations were found. For drag-and-drop the most frequently used application group was presentation editors which is comprised solely of Microsoft PowerPoint (on average 3.45 operations per hour). Spreadsheet editors (comprised of only Microsoft Excel) were the second most frequent application group observed followed by e-mail clients which comprised of Mozilla Thunderbird and Microsoft Outlook. The results show that participants performed clipboard and drag-and-drop operations in distinctly different windows/applications, possibly indicating certain applications or groups of applications are better suited to different methods of data transfer. Figure 26 shows the full results.



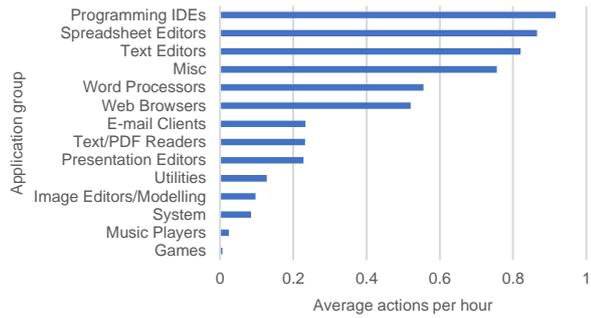


Figure 25: Average within-application (top) and between-application (bottom) clipboard operations per hour. Utilities are applications such as Windows Explorer, Calculator, and Remote Desktop. Not the different horizontal scales.

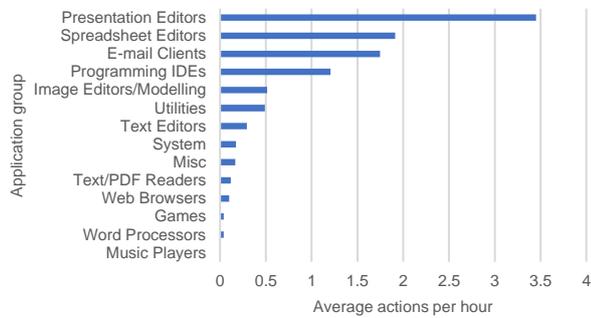


Figure 26: Average drag-and-drop actions per hour by application group.

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. Clipboard operations most commonly occurred in programming IDEs.

This thesis' data shows a similar ratio of within and between application data transfers as that reported by Stolee et al. (2009). This study's data saw 69.5% of actions within a document, while Stolee et al. (2009) reports 70% - this proportion has remained constant in the time between studies. Stolee et al. (2009) 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, this study 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 down to the group of participants in the studies - for example, a sample with non-programmers would clearly not use IDEs.

5.2.2.3 Transfer Time, Type & Size

Transfer Time: Clipboard

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

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 27. From these observations the majority of clipboard operation pairs are chronologically close (76.74% 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 operations.

Transfer Time: Drag-and-Drop

For drag-and-drop data the time taken to complete the operation was recorded from the time the mouse button was depressed until the time it was released. The mean drag-and-drop action completion time was 6.00 sec (s.d. 5.52 sec) see Figure 28. A far greater percent of actions occurring in the 0-2sec range (47.43% compared to 15.05% in copy and paste) were observed. 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.

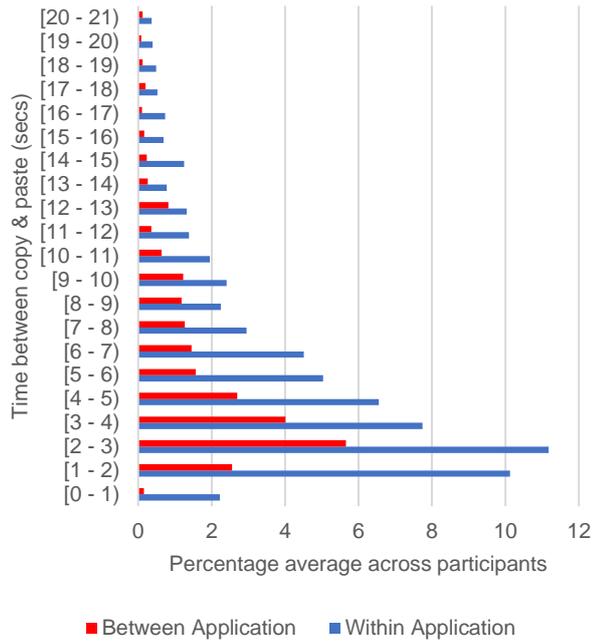


Figure 27: Distribution of times between cut/copy and paste action (average percent across participants). Graph truncated at 20sec for clarity.

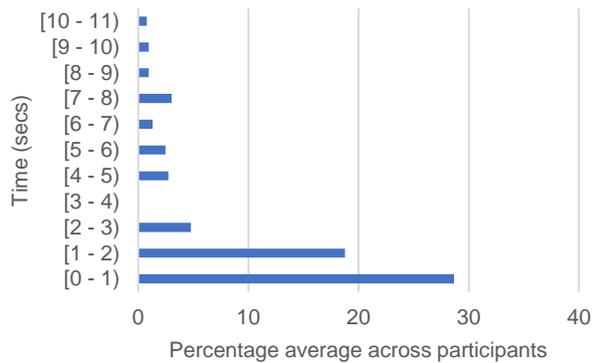


Figure 28: Distribution of drag-and-drop data transfer time (shown as average percent across participants). Graph truncated at 10sec for clarity.

Data Type: Clipboard

The type of data (and its size) transferred was recorded 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; 27% textual and 2% file operations occurred between different applications.

Of the textual data transferred by participants 62% of text strings were between 0 and 50 characters in length. 14% were between 51 and 100 characters in length (Figure 29). The movement of text data both within and between applications is reasonably well modelled by exponential decay curves (R^2 values of within=0.76 and between=0.87)

For the transfer of files, where possible (71% of occurrences), the logger recorded whether one or multiple files were transferred. 74% of the time one file was transferred while 26% of transfers involved multiple files. This thesis' study observed 61% of text transfer actions were for short strings (<50 characters); Miryung et al. (2004) 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 Miryung et al. (2004)'s findings here also apply to the broader scope of computer use.

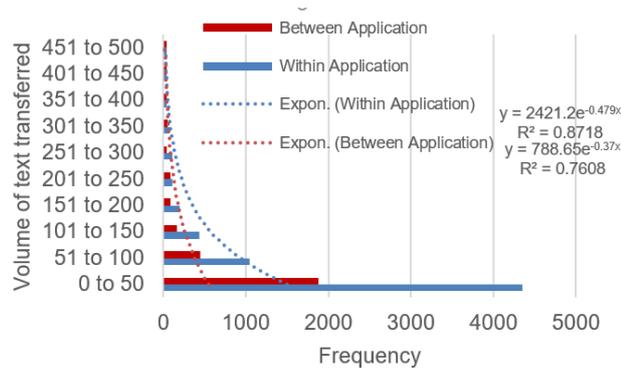


Figure 29: Length of clipboard text transferred by number of characters (data over 500 characters (10%) is not shown for clarity).

Number of Window Switches

As part of the study the number of window switches that occurred between the copy/cut and corresponding paste operation when data was transferred between different applications was recorded (Figure 30). 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 between the source and destination: this is again well modelled by exponential decay ($R^2=0.81$).

5.2.2.4 Methods of Initiation

The initiation method for all clipboard operations was recorded (Figure 31). Participants can initiate clipboard operations by using the mouse/trackpad cursor (menus, buttons, or context menus, 29.95%) or through keyboard shortcuts (70.04%).

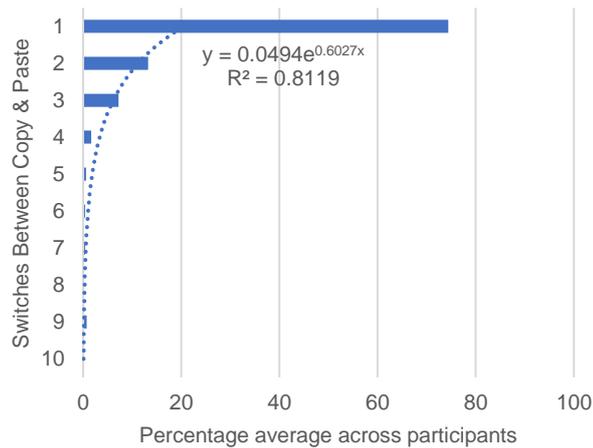


Figure 30: Number of window switches between copy/cut and paste (between-application data transfer only).

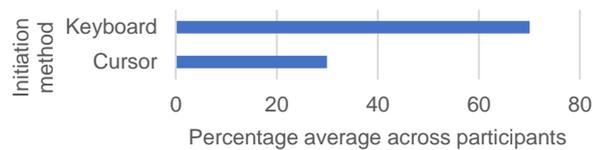


Figure 31: Clipboard operations by initiation method.

To understand if the input method changes mid-operation the method of initiation for source and destination operations was analysed. A small increase was found in the average use of the keyboard between source and destination operation (2.6%, Figure 32) 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, for drag-and-drop it was analysed whether the available pointing device (trackpad or mouse) influenced the frequency of use of this interaction (Figure 33). 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.

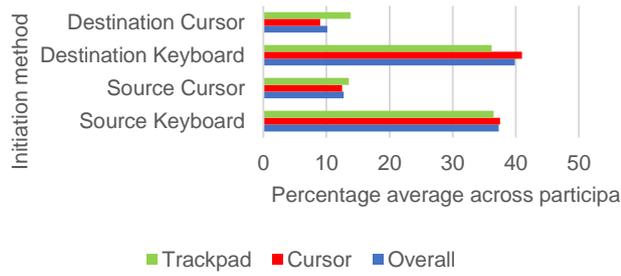


Figure 32: Clipboard operations by initiation and input method.

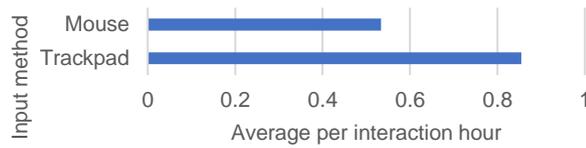


Figure 33: Average drag-and-drop operations per interaction hour by input method.

5.3 File Directory Access

Interaction with files/folders is performed on a regular basis by users of computers. As part of the understanding of user interactions the ways in which files/folders were used by participants was studied and an overview of usage is presented here.

The amount of times the Windows Explorer program was opened and the directories used within it were logged as part of the study. Data was logged not only from main Windows Explorer windows but also from “Open” and “Save” dialogs which prompt the automatic opening of Windows Explorer in these dialogs. Of all the interactions an average across participants of 69% were logged in main Windows Explorer windows. 14% were recorded in “Save” Windows Explorer dialog windows and 17% were recorded in “Open” Windows Explorer windows.

Each time a new directory was identified in each of the above Windows Explorer types the directory that was accessed was logged. The depth in the file tree where the directory was located was of interest to the research as it explains how deep in directories users are searching for or saving information. 23% of all Windows Explorer actions logged a directory 5 levels deep in the file system. 6 and 4 levels deep were the next two most

frequent directory levels logged. The deepest level logged was 14 while only 6% of actions were 1 level deep.

Analysis was also performed on the directory the user started in within the file system (the first directory recorded during a duration of Windows Explorer activity). Again, 23% of actions began at 5 levels deep in the directory tree. Only 6% of interactions began at level 1 while the highest directory depth start point recorded was 12.

Each time a file event was logged, the drive letter it occurred in was also recorded. 58% of actions occurred on drive C:\ while D:\ and H:\ were the next most used drive letters.

Frequent usage of the same directories was observed across participants with some directories being used in excess of 600 times.

5.4 Summary of Insights

This chapter presented results from the longitudinal log study undertaken as part of this thesis. The three main areas of analysis were (1) window switching, (2) data transfer and (3) file directories accessed.

5.4.1 Window Switching

Most windows are switched to using the “general window click” method.

This thesis found that the most commonly used switching method between two windows was “general window click”. This is in effect a click within the boundary of the destination window. This is the most direct method of window switching available as the user is clicking directly where they want to be. In terms of cross-application interaction, this could mean the user have a small number of windows visible in the forefront of their screens that they frequently switch between. This in turn could highlight the fact they could be part of the same task the user is performing. This insight advances knowledge on cross-application interaction by providing details surrounding how different applications are switched between.

The same window switching methods are generally used on both single- and multi-screen users.

This thesis found that the “general window click” method of switching windows was the most frequently used switching method for both single- and multi-screen users. Previous work by (Tak, 2011) showed that for single-screen users, the taskbar was the most frequently used switching method whereas for multi-screen users, the “general window click” method was more frequently used. This thesis’ results show that now single-screen users are using the “general window click” method more which could be down to the bigger screens currently available allowing users to utilise displaying “half and half” windows where one window is displayed on one half of the screen while another is on the other half. These two windows which sit side by side could be in close physical proximity to each other because they are part of the same task and therefore cross-application interaction may occur between them. This insight advances knowledge on cross-application interaction by providing details surrounding how different applications are switched between on single- and multi-screen users.

The number of windows/applications open at switch time has not changed since the previous Tak et al. (2009) study.

This thesis found that the number of windows open at switch time was generally 7 for single-screen users and 11 for multi-screen users. Previous work (Tak et al., 2009) showed similar numbers (6 for single-screen and 11 for multi-screen users). This shows that over time this data has not changed significantly. The numbers of 7 for single-screen and 11 multi-screen windows being open at switch time could indicate that these windows are part of the same task and that cross-application interaction is occurring. The Z order of switched to windows also is in most cases only 1 level deep in the window stack. This indicates that the required window to be switched to was recently used and therefore these switches could be cross-application interactions between windows which are part of the same task. This insight advances knowledge on cross-application interaction by showing that windows being switched between are recently used meaning they could be being used as part of cross-application interaction.

5.4.2 Data Transfer

The clipboard is more commonly used over drag-and-drop for data transfer operations.

This thesis found the clipboard is used for data transfer over drag-and-drop. Clipboard usage on desktop PCs is frequent (5.3 times per hour) which indicates data is being transferred between applications and as such, cross-application interaction is occurring between these windows/applications which could indicate that these windows/applications form part of the same task the user is performing. This insight advances knowledge on cross-application interaction because it proves it is occurring and goes into detail explaining how it is occurring.

Object-based interfaces encourage drag-and-drop.

This thesis found that the type of interface can encourage certain transfer methods. Those which are object-based encourage drag-and-drop over clipboard interaction. Drag-and-drop is therefore more frequently used within object-based interfaces (such as Presentation editors) and rather the more frequently used clipboard data transfer method is more frequently used in non-object-based interfaces. This could also include between application data transfer which could in turn imply cross-application interaction again indicating the source and destination window/application are part of the same task. This insight advances knowledge on cross-application interaction by providing details surrounding between application cross-application interaction which in turn is proof this interaction is occurring.

Data transfers are quick and involve text or files.

This thesis found that data transfers are generally quick and involve text or files. It is possible that participants knew where the destination window was, possibly due to the fact they were recently working on it or working within it at the same time as the source window. Again, this implies cross-application interaction applying between the source and destination windows and therefore as the destination window was possibly known

by the user, could further imply these windows are part of the same task the user is performing. This insight advances knowledge on cross-application interaction by providing data detailing the fact the destination window from a data transfer operation is known to the user possibly because of frequent cross-application usage .

Within-application data transfer occurs more frequently than between-application data transfer.

This thesis found that for both clipboard and drag-and-drop actions, the majority of actions (69.5% for clipboard and 97% for drag-and-drop) occurred within the same application. To understand in which types of applications this data transfer occurred, applications were grouped based on their primary function. Drag-and-drop's use was dominant over the clipboard in Presentation Editors (3.5actions/hour, likely due to users positioning elements on screen), spread-sheets (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). The explanation as to why more within-application data transfer occurred is down to this extensive use in these kinds of applications where data is required to be transferred around the application/window itself. This in turn does not mean that the between-application data transfer observed was not used to transfer data between two sources which are part of the same task, only that data transfer within the same application is also used in tandem with that transfer between applications which is of a cross-application nature and could therefore imply data is being transferred between windows/applications which form part of the same task. This insight advances knowledge on cross-application interaction by providing proof of within-application interaction is occurring and while not considered cross-application interaction, there is a high chance the data transfer is part of the same task if being transferred within the same application .

5.4.3 File Directory Access

Files were rarely accessed at top level directories.

This thesis observed files being accessed were generally not top-level directories. 23% of all Windows Explorer actions logged were at 5 levels deep in the file system. This could indicate users are working in “project” directories and not the desktop/root folders which could mean they are working in a cross-application way on a specific task by saving files related to the same task in the same location (possibly one task per file directory). This insight advances knowledge on cross-application interaction by providing details surrounding how users are saving files related to the same task possibly in the same locations.

Files in directories deeper than level 10 are rarely used for the storage of files.

This thesis found that users do generally not use directories deep within the file system to store their files. The deepest level logged was level 14 while only 6% of actions were 1 level deep. This could indicate that files located in very deep directory trees within the file system are hard to locate for the user. Task-related documents need to be easily located and the results show that not storing files in deep directories possibly make them easier to locate and could indicate that these files stored together in directories form part of the same task, making cross-application interaction easier. This insight advances knowledge on cross-application interaction by providing details surrounding how users are saving files related to the same tasks not too deep (over 10 levels) in the directory structure, possibly because these are hard to locate. The fact a few directory levels are seen regularly could indicate that files are being saved together because they are part of the same task/project the user is undertaking.

5.4.4 Summary

These results provided an insight into how users are performing window switch, data transfer and file directory operations and how these relate back to the cross-application interaction area (window switching increasing, transfer between applications becoming frequent and file directories not extremely deep in the file system or top-level possibly indicating files are grouped by project). However, due to the nature of logging, additional information such as the task that the user is performing is not recorded. As such, “by task” analysis cannot be conducted by this Thesis. Results were further built

upon in the next chapter as the basis and justification for the QuickFileAccess tool which was developed in order to aid users when locating files/folders. These analyses focus on the relationship between (1) file directory and clipboard events and between (2) file directory and window switch events. This provided a starting point for the algorithm which forms the functionality behind the QuickFileAccess tool. These analyses are based on cross-application interactions as that is the focus for the research. The relationship between these types of data show how different resources can be used in tandem to perform a certain task (cross-application interaction) and if a relationship is found between the two, it further re-enforces the research communities' knowledge of cross-application interaction. These further analyses are fully described in Chapter 6.

Chapter 6: QuickFileAccess

The final research question of this work seeks to understand how the efficiency of interactions in multi-application desktop environments can be improved based on the analysis and understanding of data collected. Following the development of the MultiLog tool (described fully in Chapter 3) and the analysis of the log data collected as part of the log study (described fully in Chapter 5) a tool was developed to analyse behaviour in real-time to increase the efficiency of users while completing cross-application tasks. QuickFileAccess is an extension of the MultiLog logging framework that modifies the built-in Microsoft Windows “Suggested Folders”/Quick Access folders within Windows Explorer based on relevant suggestions derived from real-time analysis of the data collected by the MultiLog system (clipboard and file directory).

6.1 Justification of Approach

Once the three areas (window switch, clipboard and file directory) of analysis has been completed (see Chapter 5), extra analysis was undertaken to better understand the relationship between these areas and to provide a justification to the development of QuickFileAccess. The analysis will aid understanding of the connections between different file directories thus assisting in the design and development of future interfaces. The use of files/folders is an extremely basic task which is employed by users on a frequent basis. The access of these files inside various directories is an interesting aspect of the ways in which participants use computers. In particular, links between these directories and their relationship with various applications are of interest. For example, if two applications access the same file, does this mean they are related? If copy and paste actions occur between two files, does this mean they are also related? If the links between files with these different interaction techniques can be proved, these results can be used to improve interfaces and operating systems or even develop certain

tools/visualisations which show proposed file directories related to the task the user is currently working on. This could improve user efficiency by allowing users to select directories where files should be saved faster by analysing behaviour and the interaction between files/applications/directories.

In order to develop the final tool analyses were performed to understand better how these file directories were utilised by users. The data analysed was that collected as part of the main longitudinal log study described in Chapter 4. Using this data two distinct analyses were performed:

1. The relationship between copy and paste actions and file directories accessed.
2. The relationship between window switch actions and file directories accessed.

For (1) work was undertaken to understand better the connection between copy and paste (clipboard) actions and file events. For each clipboard event recorded the application the action was performed in was logged and for each application recorded file events within one minute were collated and added to a collated output list for later presentation in the QuickFileAccess tool. This output showed file directory access events in close proximity to clipboard events indicating that this could be used as a measure of the relationship between the two types of data.

For (2) a similar analysis was conducted. Each window switch was extracted and the application switched to was logged. For each application another list was collated containing any recorded file directory events within one minute of the window switch/application data. This list shows window switches in close proximity to file events and will also be used in the output presented to the new QuickFileAccess tool.

The tool uses the approach in (1) as the relationship between clipboard and file directory operations because it provides a better link between applications being part of the same task than window switching does (as windows which are not part of the same task can be switched between) but the transfer of data between two applications shows some direct connection between them.

This chapter firstly explains the context of the QuickFileAccess tool and explains what the Windows Explorer Quick Access list is and how it is used. Secondly, a full description of the QuickFileAccess tool is provided exploring initial algorithms used to manage the list before fully explaining the selected algorithm which was used in the

pilot study. The pilot study is then fully described along with the results (both quantitative – from log data, and qualitative – from user feedback). A brief discussion of the QuickFileAccess tool is then presented.

6.2 Context

Windows Explorer is the primary application used on Microsoft Windows operating systems to graphically navigate around files and folders stored on local or remote file systems. Windows Explorer has been included (pre-installed) on releases of Microsoft Windows operating systems since Windows 95. In later versions of Windows Explorer additional functionality beyond the simple browsing of files and folders has been added. In the Windows 7 version “Libraries” were introduced. These “Libraries” aggregate files/folders from various different locations, local and remote. Searching within a “Library” queries both local and remote systems as appropriate.

A new feature present in later versions of Windows Explorer is the addition of “Suggested Folders”/“Quick Access Folders”. This group of files/folders appears in the left hand tree structure under the heading Quick Access and also appears when the Quick Access button is clicked in the left hand panel and in many versions when Windows Explorer is first opened from the task bar/executable file.

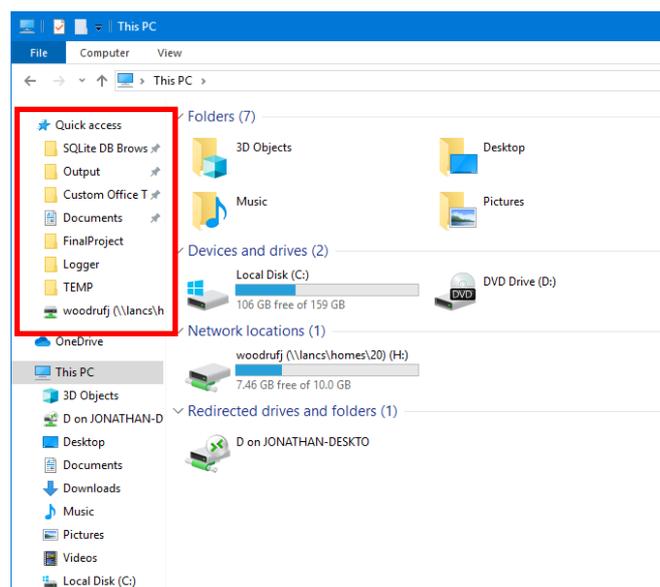


Figure 34: Windows Explorer "This PC" view (Windows 10). Quick Access list is highlighted in red.

The left hand panel shows (under the Quick Access heading) the list of frequent folders navigated to by the user (WindowsTechies, 2019). These are also duplicated in the main Windows Explorer panel when the Quick Access button is clicked. The centre panel also shows a list of frequent folders (shown in a blue box) at the top. Underneath this a list of recent files exists (shown in a red box) (Figure 35).

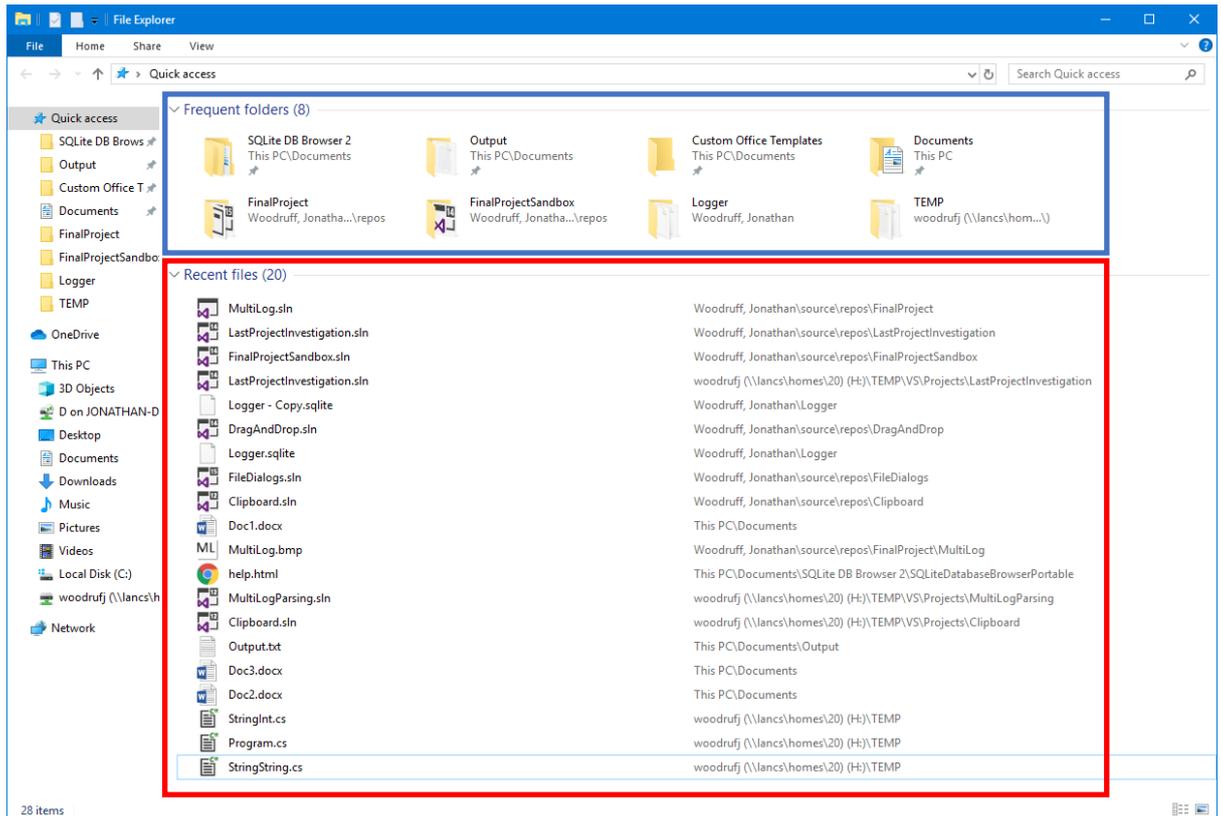


Figure 35: Windows Explorer Quick Access view (Windows 10).

If one of the Quick Access recent files is clicked the file will open in the appropriate application. If a frequent folder is selected from either the main panel or the Quick Access list in the left-hand panel this will re-direct the Windows Explorer window with current focus to the directory clicked. This is a quick way to access most frequently used folders within Windows Explorer. Users can “pin” folders to the Quick Access list. This ensures they remain on the list continuously. The list is updated to reflect the current most frequently used/accessed folders by the Windows Explorer process.

6.3 QuickFileAccess Concept

Results from the previously described longitudinal log study (Chapter 5) help inform the decisions for the development of the QuickFileAccess tool. Clipboard data clearly shows a direct connection between two applications and this type of interaction is a good indicator that two applications (the source and destination of the clipboard operation) are related.

Using this, the QuickFileAccess tool will dynamically manage the Windows Explorer Quick Access list to populate it with file directories believed to be relevant to the cross-application task at hand by analysing clipboard and file directory log data using an algorithm described fully below.

6.4 Description

The QuickFileAccess tool is an extension to the previously described MultiLog logging framework (Chapter 3) which uses an algorithm (described later) to present relevant file directories to the user through the Windows Explorer Quick Access list. The development of QuickFileAccess hopes to aid user efficiency during the performance of tasks requiring multiple applications on Microsoft Windows machines by making file access more efficient.

A characterisation of the ways in which window switching, data transfer and file access interactions are performed is presented within Chapter 5 of this thesis. Section 6.1 aims to perform further analyses to gain an understanding of the connections between these different types of interaction. This analysis is the basis for the development of QuickFileAccess which uses a conjunction of clipboard and file directory events to identify and present relevant directories to the user. This methodology was chosen because clipboard operations by nature are cross-application actions. Folders are also accessed by multiple applications sometimes to complete the same task. The alternative algorithm was to use a combination of window switches and file directory events but, due to the nature of window switches, it is impossible to tell if two windows switched between are related or whether the user was in fact switching tasks when the switch was initiated.

The QuickFileAccess system runs within the main MultiLog process and is active whenever the MultiLog program is running. It analyses (using the offline analysis code presented in Chapter 5 integrated into the MultiLog system) in real-time the data collected using the MultiLog system and interfaces with the previously described Quick Access list to show more relevant folders in order to assist the user to become more efficient when undertaking cross-application tasks by allowing them to find more relevant directories faster.

The functionality of the QuickFileAccess software is two-fold. Firstly, the QuickFileAccess system has to analyse the clipboard and file directory data collected. Secondly, it uses this output to dynamically manage the Windows Explorer Quick Access list. The most important part of the system is the algorithm it uses to add items into the Windows Explorer Quick Access list. Various different algorithms were considered before the one believed to be the most relevant was selected and implemented in the QuickFileAccess system.

6.5 Algorithm Development

As mentioned earlier various algorithms using the relationship between copy and paste actions and file directories and also the relationship between window switch actions and file directories were considered. The first option was chosen because the action of copy and paste by default shows transfer of data between applications which shows a direct connection between them. Switching windows between applications may or may not show a direct connection between them (depending on whether the switch occurred corresponding to switching task).

Algorithms concerning the relationship between copy and paste actions and file directories were considered. Firstly, a simple solution was proposed and tested. This took all file directory events and simply added them into the Quick Access list if not already present. This solution was tested and it was realised this would require integration with clipboard or window switch events to become a more stable, reliable algorithm due to the high volume of directories being added and therefore the user being unable to locate relevant ones with the list changing so frequently. It was decided that,

due to the cross-application nature of clipboard operations, an algorithm that attempted to match directories to clipboard operations would be more relevant.

Secondly, an algorithm which obtained all clipboard events and attempted to match a cut/copy to a corresponding paste action was designed, coded and tested. When this was completed the application the event occurred in was extracted and any file directory events within one minute of the clipboard data were added to the Windows Explorer Quick Access list. This algorithm tries to make the connection between the file directory data and clipboard data (to prove cross-application interaction has occurred). This solution sometimes resulted in more than one file event being added per clipboard operation and it was realised the algorithm would have to select the more applicable file directory to add to the Windows Explorer Quick Access list in order to be more accurate.

Thirdly, a refinement of the second algorithm was developed. File directory events within one minute (to prevent CPU overload) were still extracted except one event (the one closest in time to the clipboard event) was extracted and used to add to the Windows Explorer Quick Access list. This way only one item was added and this was deemed the most appropriate due to its time proximity to the corresponding clipboard operations.

After testing with the third refined algorithm above and in conjunction with the data which was available for analysis, it was found that if the two paired clipboard events happened in Windows Explorer, file events within five seconds of the event would be obtained and attempted to be matched to the clipboard data. This would show a direct connection as the clipboard data would show the item was cut/copied/pasted from/to the directory which would also have its own file directory entry. This matching of the directories in both the clipboard and file directory data shows a direct connection between the two. If a match occurred it was added to the Windows Explorer Quick Access list.

Also, if analysis of the pair of clipboard operations occurred and showed a between application copy and paste (copy from Word, paste into Excel for example) there is a chance a file directory event from a Windows Explorer Open or Save As dialog box would have occurred. Consequently, for these events the first Windows Explorer Open or Save As dialog event was extracted as was the window switch before it. The window switch was then matched to the clipboard application and if a match occurred, the file directory was added to the Windows Explorer Quick Access list.

6.6 Final Algorithm

The final algorithm attempts to match clipboard actions to file directories to show a direct connection between clipboard and file directories accessed. A full description of the final algorithm used in the pilot study is below:

Once every minute (justification of time explained after the algorithm):

1. Extract all logged clipboard operations.
2. For each clipboard operation, match cut/copy operations to paste operations and vice versa.
3. Identify the source and destination application cut/copied/pasted to/from.
4. Determine whether *within application* Windows Explorer interaction has occurred or whether *between application* interaction has occurred.
5. If *within application* Windows Explorer interaction has occurred:
 - a. Get file events (an event generated when a Windows Explorer window is navigated to a different directory) within five seconds of the clipboard event and attempt to match the file directory from the clipboard event to that of the file event.
 - b. Add any matches from (a) to the Quick Access list.
6. If *between application* interaction has occurred, the following steps are executed for both the source (cut/copy) operation and the destination (paste) operation:
 - a. Search log lines for any Windows Explorer “open” or “save as” file dialogs which have appeared prior to the clipboard cut/copy and paste events.
 - b. For each of the file dialog events in (a), find the window switch immediately prior to that event.
 - c. If the window switch was to the application the clipboard operation occurred in, add this file directory from (a) to the Quick Access list.
7. If the number of items in the Quick Access list is greater than eight, the item at position zero is removed from the list. This is to prevent overload of the Quick Access list.

The time period of one minute was chosen as the best solution to a time that updated the Quick Access list regularly but without impeding the resources of the running operating system which would affect users when interacting with their computers.

6.7 Results Presentation UI

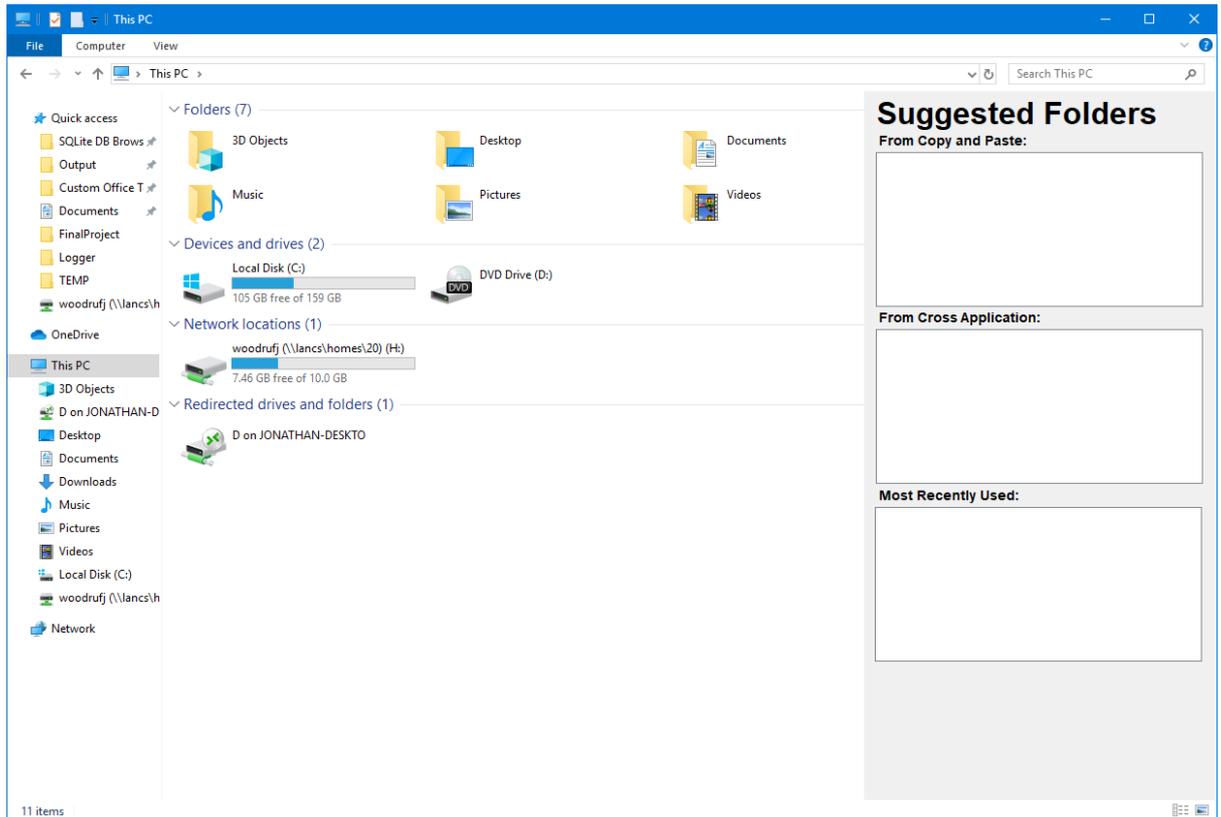


Figure 36: Original "Suggested Folders" Windows Explorer overlay.

The results identified by the QuickFileAccess software are presented back to the user. Originally, an overlay to the window was considered (Figure 36). However, this would add additional controls to the Windows Explorer window which may cause the Windows Explorer interface to become more complex to interact with. Instead, the relevant file directories identified by the analysis stage are presented in a list internal to the QuickFileAccess system which are then added to the Windows Explorer Quick Access list. This enables the user to interact with them in the same way they do with items added by the Windows Explorer process. Clicking any items that QuickFileAccess adds to the Quick Access list will re-direct the open Windows Explorer window to the directory which has been clicked by the user.

6.8 Algorithm Limitations

During the development of the final algorithm it was realised it has a small number of limitations. For example, the file directory MultiLog “plug-in” does not collect data from the “new style” Microsoft Office Open/Save screens. Events are only logged when users fully browse for the required file in the standard Windows Explorer “Open”/”Save As” Dialogs. This is due to the MultiLog software being developed prior to this new feature becoming available in Microsoft Office. Also, on a small number of occasions the logger may miss events or duplicate events for reasons outside MultiLog’s control (within Windows APIs which are used to query data).

6.9 Beta Testing

Once the software was developed it was tested by three beta testers for a period of four weeks. All three users were advanced users and utilised their machines for long periods of time and so were ideal testers for the early versions of the software. During the four week period various bug fixes and enhancements to the algorithm were made in order to give the best possible chance of presenting relevant directories to end users. The aim of the QuickFileAccess pilot study was to evaluate its usefulness and effectiveness as a tool aimed at increasing user efficiency by decreasing the amount of time users spend locating required directories.

6.10 Pilot Study

The pilot study aimed to gather quantitative feedback on the use of the modified Windows Explorer Quick Access list through logging interactions with it. It also aimed to gain qualitative feedback from users surrounding its usefulness and whether they felt they could locate folders/directories easier with the QuickFileAccess tool. The analysis focussed on five main points centred on the ways in which the Quick Access list in Windows Explorer was managed and interacted with. These areas are:

1. The frequency of clicks on items in the Quick Access list. This shows an indication of how frequently the Quick Access list is used and therefore whether it is a good place to present relevant directories to the user.

2. The frequency at which items were added to the Quick Access list. This shows the rate at which the QuickFileAccess tool added events to the list and therefore how many relevant directories it found within the log data.
3. The frequency at which items were removed from the Quick Access list. This shows whether the correct maximum number of items was selected (8) and if frequent removal is required, whether a space larger than the Quick Access list would be more useful to users.
4. The content of the Quick Access list at various times during the study. This allows monitoring of the contents and tracking the number of items present which helps indicate whether the Quick Access list was a reliable choice of location to present relevant directories.
5. Qualitative feedback from users regarding their experiences with the modified Quick Access list. This allows qualitative feedback to be analysed and for the end user to rate their use of the Quick Access list, whether they believed more relevant file directories appeared there and whether they felt they become more efficient when using the QuickFileAccess tool.

Following the beta testing period the software was deployed to nine users for a period of three weeks. Three of the participants were from the academic environment, four were business users and two used their computers solely for personal/home use. Seven of the participants were employed while one was a MPhil student and another retired. Of the participants six were male and three female. Participants were sent a demographic survey prior to the study commencing and the software was installed on their computers for them to use for the period. On conclusion of the pilot study, participants completed a post-deployment survey. On average, the percentage of days users has some kind of interaction (measured by window switches) across the pilot study was 39.15%. The highest number of days with interaction during the pilot study was recorded was 16 (2 users) and the lowest number was 3 (3 users).

The following section reports the results of the analysis of the pilot study used to evaluate the QuickFileAccess tool.

6.11 Pilot Study Results

This section explores the results generated from the analysis stage of the QuickFileAccess pilot study. The sections reference the relevant analysis areas listed previously.

6.11.1 Frequency of Items Clicked in the Quick Access list

This generally shows that an extremely low number of clicks on the Quick Access list occurred. During the three weeks for which the pilot study ran, the number of times items were clicked across all users was 40. The average number of clicks on the Quick Access list across participants was 4.4 (s.d. 5.12) which indicates low frequency of use. Figure 37 shows a breakdown of the number of times each participant clicked on items in the Quick Access list throughout the pilot study.

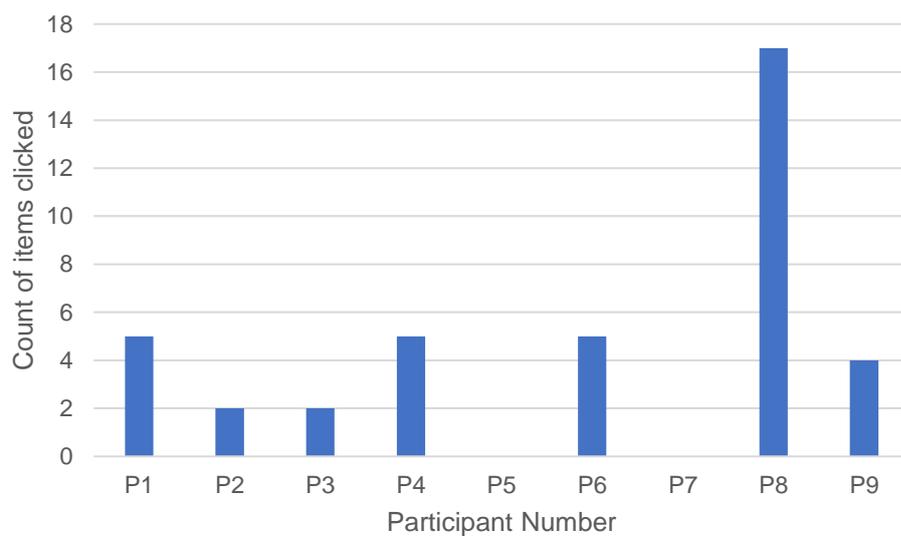


Figure 37: Breakdown per participant of the number of times items in the Quick Access list were clicked.

Of the items clicked it was found that across all participants none were added by the QuickFileAccess tool and as such all were added by Windows.

6.11.2 Frequency of Items Added to the Quick Access list

It was found that a total of 56 items were added to Quick Access lists by QuickFileAccess across all participants with an average of 6.2 (s.d. 5.26) across participants. Figure 38 shows a breakdown of the number of items which were added to the Windows Explorer Quick Access list by participant.

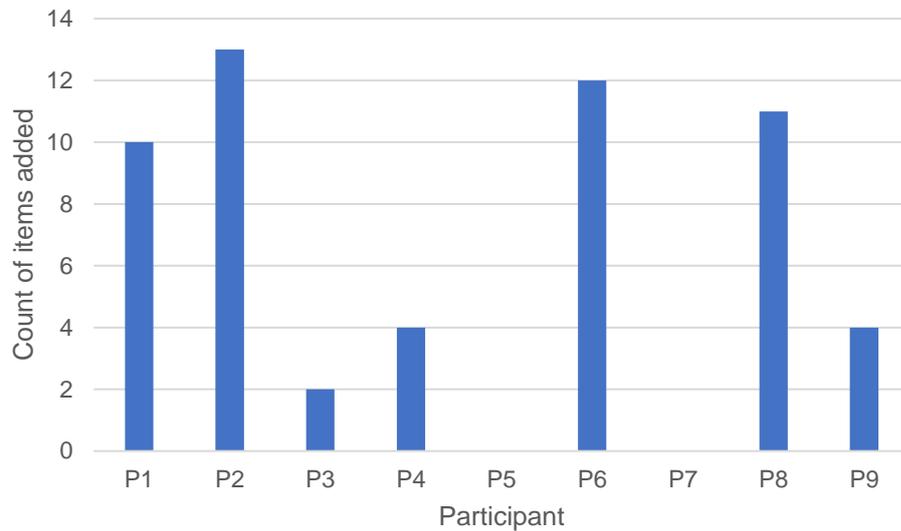


Figure 38: Breakdown of directories added to the Windows Explorer Quick Access list by QuickFileAccess across participants.

6.11.3 Frequency of Items Removed from the Quick Access list

Across the period of three weeks during which the pilot study ran, a total of 28 items were removed from the Windows Explorer Quick Access list. This was necessary when the list was full so as not to overload the user with high numbers of directories in the list. An across-participant average of 3.1 (s.d. 3.29) items were removed during the pilot study.

6.11.4 Content of the Quick Access list

The content of the Quick Access list was recorded at various times during the study. The list in its entirety was recorded every time a click, add or remove action was performed. The total number of unique items displayed in the Windows Explorer Quick Access list (the sum of all items across all times the contents of the list was recorded)

was 500 with an average of 55.5 (s.d. 47.2) items across participants. Figure 39 shows a breakdown across participants of the number of items displayed in the Quick Access list.

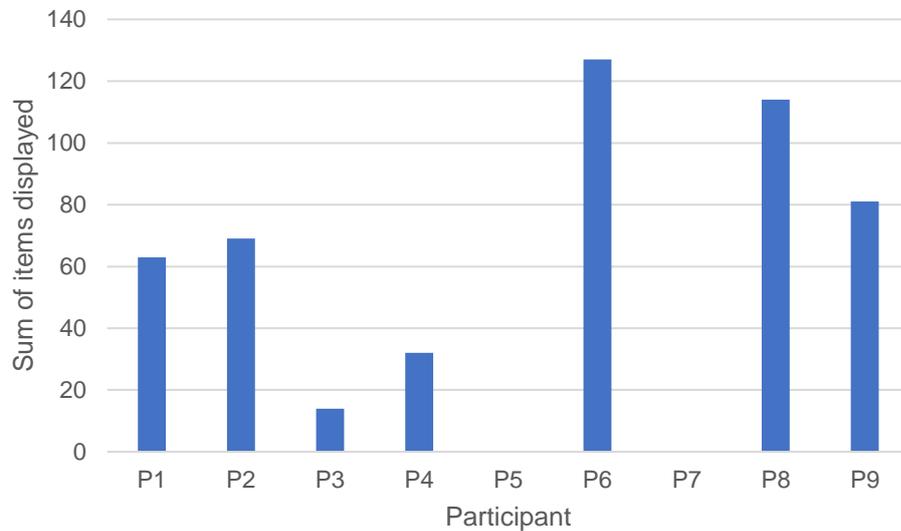


Figure 39: Breakdown of sum total number of items displayed in Windows Explorer Quick Access List.

The number of items which were present in the list when it was recorded (every time the tool added or removed an item and every time one was clicked by the user) was also logged. The most frequent number of items present in the list was 8. The QuickFileAccess tool limited the number of allowed entries to 8 which would suggest that users' Windows Explorer Quick Access lists were full regularly. A count of 9 items was recorded twice during the pilot study. This could suggest Windows added additional items which were not subject to the number limit specified within QuickFileAccess. Figure 40 shows the full results.

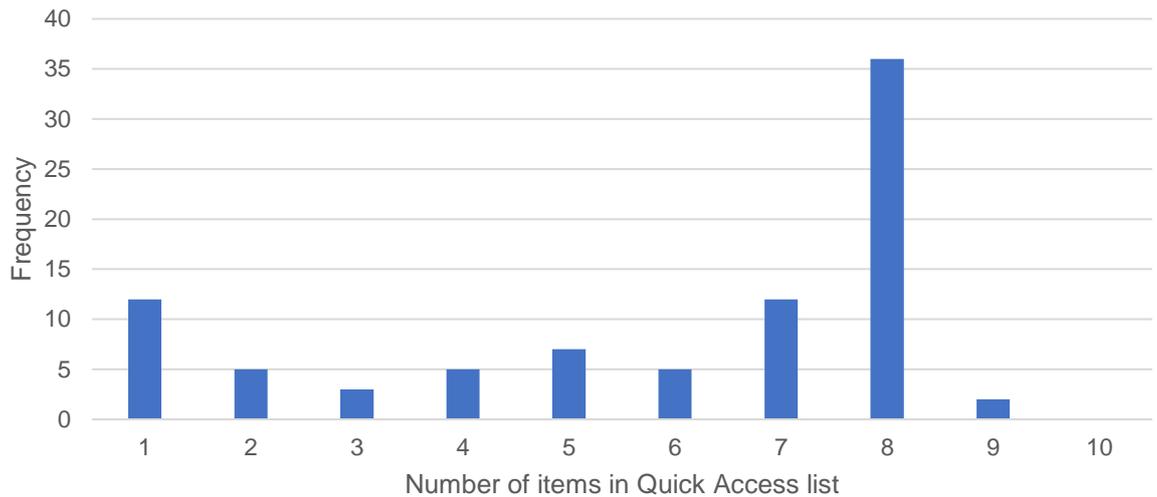


Figure 40: Frequency of number of items shown in Windows Explorer Quick Access list when queried.

6.11.5 Qualitative Feedback from Users

Two surveys were carried out as part of the pilot study; one prior to the pilot study starting and one on completion of the pilot study. The survey asked users to rate their Quick Access use before the pilot study began. Various responses from 0 (not at all) to 10 (a great deal) were reported. Figure 41 shows the ratings of each participant.

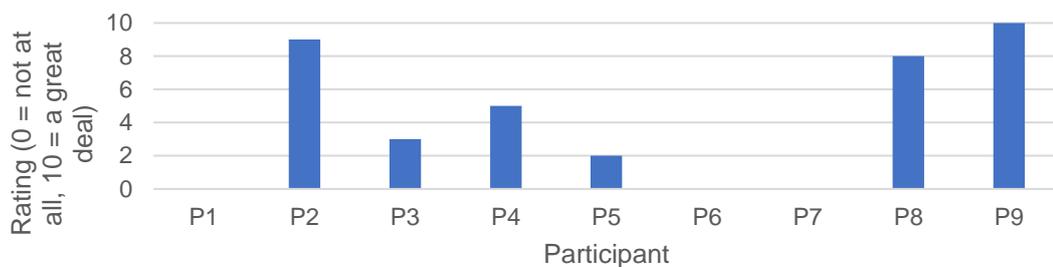


Figure 41: Overview of participant's Quick Access usage prior to pilot study.

Following the pilot study, participants were asked to rate their responses to four questions:

1. I noticed different directories appeared in the Quick Access list to normal.
2. The file directories that appeared in the Quick Access list were more relevant to me.

3. I used the Quick Access list more because file directories which appeared there were more relevant to me.
4. The tool improved (decreased) the amount of time I generally have to spend looking for files/directories on my computer.

Most participants ranked responses high. A rating less than 3 was not recorded by any participant and most ratings were over 5. Figure 42 shows an overview of participant responses to these four questions. Ratings are from 0 (strongly disagree) to 10 (strongly agree).

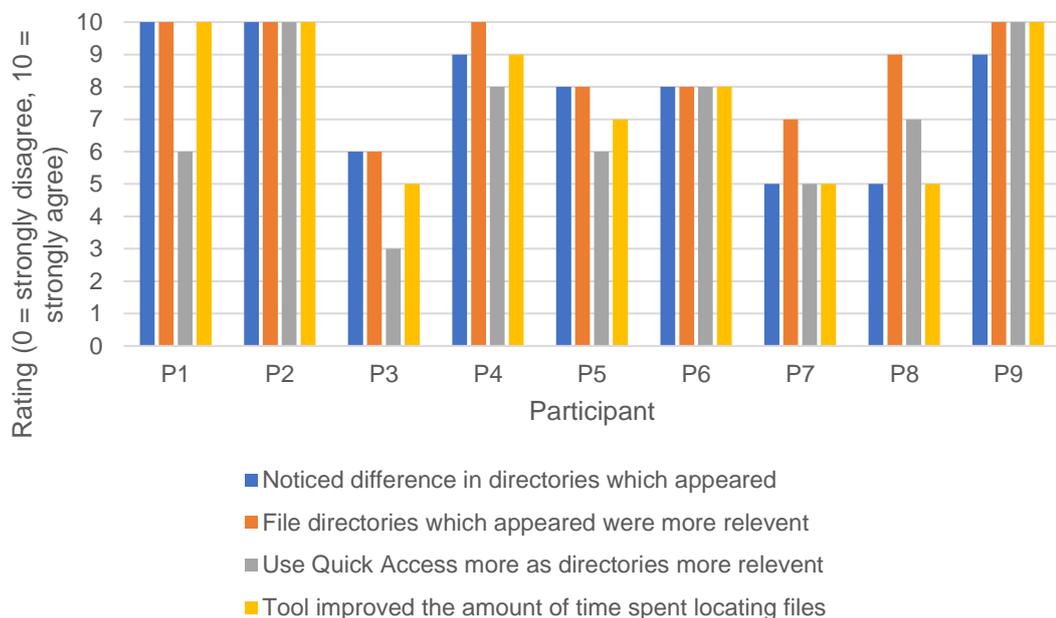


Figure 42: Participant responses post-pilot study.

6.11.6 Summary of Results

The analyses of the data collected during the QuickFileAccess pilot study have provided useful insights, particularly into the use of the Windows Explorer Quick Access list. Results show a low number of items which were both clicked on and added to the Windows Explorer Quick Access list. This shows generally low use of the list. Few items were removed from the list with an average of 8 items present in the list at times when its contents were queried. While quantitative feedback from the log files shows low usage of the Windows Explorer Quick Access list, qualitative feedback indicates users saw a difference in the Quick Access list to normal. This could however be

because they were made aware of the Quick Access list for the first time as part of this pilot study. Due to the low frequency of events recorded this may indicate users did not know about the Quick Access list before the pilot study and therefore did not have sufficient time during the pilot study to fully utilise it. This is partially reflected in the qualitative data recorded as part of the two surveys undertaken. Although qualitative feedback was positive (and quantitative data showed low frequency of use) there is not currently sufficient data to prove the usefulness of the QuickFileAccess tool due to the restraints (time, length, number of participants, participant use of the Windows Explorer Quick Access list) of the pilot study. These restraints are the length of time for which the pilot study ran, the number of participants involved and the types of users (novice/expert).

6.12 Pilot Study Discussion

The results of the pilot study suggest low frequency of use of the modified Windows Explorer Quick Access list. Some users never clicked on items while all items which were clicked were added by Windows Explorer. As mentioned previously this could be because users were not aware of the Windows Explorer Quick Access list prior to the pilot study starting. Consequently, their interactions with it were low in frequency. This seems to indicate that the Quick Access list is not a good solution of where to present relevant file directories to the user due to its low frequency of use. This has wider implications for interaction design and it is important interface designers take note of the results indicating low usage of the Quick Access list, possibly hinting other ways of displaying Quick Access items need to be reviewed in further developments of operating systems.

The reasons why users did not utilise the Quick Access list very frequently would be relevant to further work in this area to understand whether it is a useful feature to be included with future versions of Microsoft Windows operating systems. On reflection, further guidance could have been given at the start of the study explaining how to use the Quick Access list to gain the most from it.

Another consideration to be made is whether the most relevant file directories were presented within the Quick Access list? Further analysis would need to be carried out

and alternative algorithms tested in order to ascertain which would be the best algorithm to implement into the QuickFileAccess system to increase user efficiency.

There are many directions of future work in this area which could mainly concentrate on testing other types of algorithm, such as refining the clipboard/file directory algorithm used in this thesis to present further relevant directories to the user. Other types of algorithm could also be tested including logging file directory access in more depth and possibly testing some window switch/file directory related algorithms.

If future work were to be carried out in this area, the location of identifying whether the Windows Explorer Quick Access list is the most relevant place to present the findings of the algorithm to the user due to its low use rate during the pilot study would be undertaken. Future work would also focus on alternative plug-ins to Windows Explorer or even a brand new application which users could utilise to browse to relevant directories (the next generation of “Explorer” applications) and their evaluation to ascertain a better solution of presenting the data back to the user.

6.13 Pilot Study Strengths/Weaknesses

The pilot study undertaken has various strengths/weaknesses. The pilot study explores the usage of the Quick Access list within Windows Explorer. It showed a low frequency of use indicating it is not the most relevant place to display this information to the user. The pilot study provides a contribution to the research community in that the Quick Access list may not be the best location for this information and is generally infrequently used. Therefore alternatives will need to be considered for future work such as additional interfaces or different Windows Explorer plug-ins. Interface designers can also use this information to inform the design of future interfaces taking note of the infrequent use.

There are weaknesses also to the study. For example, no control group was used so a comparison of the usage of the Quick Access list before and after the algorithm changes was not possible. This would be vital in future work to ascertain the effect of the algorithm on populating the items in the list. It is also not clear whether the quantitative or qualitative data is more reliable as both contradict each other to a point. As such, clear conclusions cannot be drawn. Finally, as a pilot study this was not a full

deployment and did not run for sufficient time or with sufficient participants to gain a full understanding of the interaction or algorithm effectiveness.

A revised study could address most of the weaknesses described here. For example, if the pilot study were to be run again, it would ideally be run as a full study, covering an extended period of time (possibly upwards of 90 days similar to the longitudinal log study conducted earlier which aimed to understand user interactions. If this study were to be run, a control group would be used so comparison between those with the new system and those without it could be made, ultimately enabling clearer statements on the usefulness of the tool to be made when comparing these two groups. A broader range of participants would be used, comprising non-academics as well as people of differing age groups and possibly those using different operating systems. The size of the participant cohort would also be increased so a better understanding can be gained from the data. The results could be compared with systems of similar functionality to see whether improvements have been made.

On the technical side of the tool, the ability for different users to use different algorithms or for the algorithms to be changed on the fly could be built in so that feedback from a varying number of algorithms can be gathered. Subgroups could also be used with different subgroups using different algorithms (if the number of participants signed up for the study was sufficient enough). As previously mentioned, analysis would be performed into the best place to present the recommended file directories to the user. Various different solutions could be proposed to users and feedback gathered to ensure that the recommendations are integrated into the existing operating system at a useful point to the user. Again, different solutions could be tried either one after the other or different subgroups of the participant cohort using different solutions with a comparison made at the end between different solutions and the control group.

In summary, the study would be scaled up considerably in order to gain a deep understanding of (1) the best algorithms to use to recommend files and (2) the best place to display the results to the user.

Chapter 7: Discussion and Future Work

Cross-application interaction is a commonly performed activity in modern operating systems that allows users to complete a certain task with the use of a subset of applications. Firstly, the work presented in this thesis provides researchers with a solution for collecting data on multi-application use in unmodified desktop environments through the use of the MultiLog software. Secondly, it also provides a deep understanding of how and when this multi-application use occurs in desktop environments (within Chapter 5) focussing mainly on window switching, data transfer and file operations. Thirdly, it provides the starting point to improving the efficiency of interactions in multi-application desktop environments by supplying the QuickFileAccess tool which analyses in real-time the data collected by the MultiLog logging framework and presents results back to the user through the dynamic modification of the Windows Explorer Quick Access list.

Researchers can use the work described in this thesis to further analyse and improve cross-application interactions in desktop environments. New studies using a different range of logging applications plugged in to MultiLog may be run to find other ways in which users are performing cross-application interactions while performing tasks on their computers. This collected data may again be analysed and work published to further increase the research community's current understanding of cross-application interaction methods. Finally, the results analysed may present opportunities in the further development of plug-ins/add-ons to well-used applications such as Microsoft Office and integrated programs like Windows Explorer which may be able to aid users in increasing their efficiency of interaction. The findings may also help inform the designs of future operating systems themselves to aid this user efficiency.

This chapter focusses on a higher level view of the work presented in this thesis and describes the progress that has been made toward answering the research questions posed in Chapter 1. It also provides important lessons for practitioners, discusses the ability to generalise the results presented within the thesis and looks ahead to possible future directions of this area of work to benefit the research community as a whole.

7.1 Progress on Research Questions

The over-arching goal of this thesis was to understand and improve cross-application interaction for users of desktop computers. The thesis set out the following research questions in Chapter 1:

- 1 How can we collect data on multi-application use in unmodified desktop environments?
- 2 How and when does multi-application use occur in desktop environments?
- 3 How can we improve the efficiency of interactions in multi-application desktop environments?

7.1.1 How can we collect data on multi-application use in unmodified desktop environments?

The first research question was considered to be complete when the MultiLog logging framework was ready for deployment for the corresponding log study. The first part of answering this question involved an overview of available logging applications and deciding if these pre-existing applications were sufficient for use in collecting the types of data required in order to learn more about cross-application interaction. After an exploration of the current logging software available for this case it was deemed that more relevant data could be collected with the creation of bespoke logging software application. Furthermore, it was envisaged that if proposed logging software is designed in such a way that existing logging applications could be plugged-in and one universal log format was presented, this might deliver another advantage to the research

community. As a result of this analysis the MultiLog logging framework was created and released to the research community. It allows the output from various pre-existing logging applications to be merged, on-the-fly, into one central log database. Loggers are configured in MultiLog's add logger wizard or through the use of the configuration file for research deployments. Various other small logging applications are provided by default within the MultiLog logging framework (for example the logging of window switching, Windows clipboard, drag-and-drop and file directory access) and can be run without further configuration. The system would then be used to assist with answering the second research question.

7.1.2 How and when does multi-application use occur in desktop environments?

The second research question was considered to be complete when the analysis stage of the collected data was complete. On completion of the 90 day longitudinal study the collected data was analysed in order to draw conclusions on user behaviour/interaction methods. Chapter 5 describes a deep analysis of three main areas of interest to this research: (1) window switching, (2) data transfer and (3) file directories. Within (1), an understanding of window switching methods, the position and size of windows and the number and state of windows/applications at switch time were presented. Focus on between application switching is prominent as this links strongly with the over-arching research goal, exploring cross-application interaction. Within (2), a deeper understanding as to the ways in which users transfer data both within and between applications is presented. The frequency of data transfer, on-screen location and source/destination applications, transfer time, types and sizes and methods of initiation are presented. The two types of data transfer studied are the use of the Windows Clipboard and drag-and-drop usage. Within (3), the ways in which users interact with file directories on the file system is provided. Links between file usage and window switches/clipboard usage are then made which paves the way to answering the third and final research question addressed by this thesis.

7.1.3 How can we improve the efficiency of interactions in multi-application desktop environments?

The third research question was considered to be complete on review of the QuickFileAccess Windows Explorer addition. Once a deep understanding into various types of interaction was gained from the results shown in Chapter 5, a tool that would aid the user's efficiency while performing tasks was developed. The QuickFileAccess tool was released to a small number of users during a short-term study to analyse its usability. The QuickFileAccess tool presents relevant directories to users through the Windows Explorer Quick Access list by dynamically managing this list. The quantitative feedback showed a low frequency of interactions with the Quick Access list however qualitative feedback from users was positive with users praising the system stating that in most cases the time taken to locate required files/directories was reduced with the aid of QuickFileAccess. There are many areas of future research into such tools, including refining the algorithm or presenting information to the user in different ways.

7.2 Lessons for Practitioners

The content of this thesis has prompted several lessons for practitioners that may be applicable in future research in this area:

- 1 *Many logging applications have distinct log formats which can make collection of data from various different loggers simultaneously difficult.* Most logging applications available in both the research community and the commercial community have separate, distinct log file formats. This makes it difficult to run a research study, for example, collecting data on various different interactions at once. The creation of a universal logging framework of which plugging in existing logging applications is easy has enabled us to solve this issue. A tool such as MultiLog can be employed to manage the data collection exercise and present results in a more universal way, making analyses easier to perform on conclusion of any study run. A single, universal log format used by all would be the ultimate solution to this problem which would make it easier and quicker for the analysis of data to be completed across many different logging systems. It

would also aid cross-topic research by standardising the ways in which logs are produced/analysed.

- 2 *Many logging tools have limitations in reliability and performance.* Many logging applications currently available to researchers have limitations in their data collection abilities. Certain loggers which provided valuable interaction data were difficult to interface to from the MultiLog framework and others sometimes provided incorrect log entries (which had to be filtered out) or used a high amount of processing power which proved to be a burden on users. Researchers would benefit from a platform which is more reliable, more efficient and more open-platform.
- 3 *The analysis of large data sets can be challenging.* During the 90 day longitudinal log study thousands of log lines were collected, some of which were erroneous. The process to filter out data which was not accurate took time and various validation tests to ensure inaccurate information had been correctly identified and removed. The standardisation of log data or the provision of tools which make it easier to filter/remove inaccurate data would be valuable to researchers in this area and in the general community as a whole and would enable researchers to analyse data faster and present valuable results in the form of papers easier.
- 4 *Analysis of current interactions informs future design of user interfaces.* The data collected by MultiLog in the longitudinal log study and the observations and conclusions made from this collected data are important points to take into consideration when designing future interfaces. The manner in which users perform window switches, data transfer and file directory access operations provides important information to take into consideration when designing future interfaces for the Microsoft Windows operating system. For example, the future design of the Windows Explorer application could provide suggestions of relevant folders through the analysis of data collected by the MultiLog logging framework and the analysis of said data by the QuickFileAccess tool.

- 5 *Users would benefit from user interface enhancements showing relevant directories which make their tasks easier to complete.* Chapter 6 describes the QuickFileAccess tool which dynamically manages the Quick Access list within Windows Explorer presenting relevant directories to the user. Qualitative feedback from the QuickFileAccess tool is positive and many users said that the time taken to locate files was reduced even though qualitative log data showed a low frequency of use of the Quick Access list. Users also reported they noticed a difference in directories which appeared, that more relevant directories were shown and that they used the Windows Explorer Quick Access list more because the directories were more relevant. Due to the low frequency of Quick Access events logged this could suggest that users were not aware of the Quick Access list before the pilot study and the period of three weeks was not sufficient time for them to get used to utilising it.

7.3 Research Generalisability

The research presented in this thesis has focused primarily on window switching, data transfer and file directory access in a single operating system. The work is however generalisable and easily extendable both within and beyond this domain. This section of the thesis outlines three areas: (1) the use of MultiLog to collect vast amounts of different data relating to many different actions/interactions, (2) the use of the data collected through MultiLog to present findings on the usage of various different interaction techniques to enable cross-application working and (3) extending the feedback of these analyses to users in different ways.

7.3.1 MultiLog

The MultiLog logging framework was employed during the study to collect data on window switches, data transfer and file directories. Due to MultiLog's design almost any other logging system that generates a log file, an XML/HTML file or a TCP/UDP stream can be plugged-in to MultiLog, enabling the data collected by such loggers to be included in MultiLog's output database. As a result of this feature there is no restriction

on the amount of data that can be collected and analysed. MultiLog can be extended easily by the addition of such loggers either through the add logger wizard interface inside the application or through the use of a configuration file for deployments. Researchers are also able to extend the MultiLog logging framework themselves if they wish. More internal loggers can easily be written and added to MultiLog if required. If other researchers consider enhancing the MultiLog logging framework with more internal loggers, this will boost MultiLog's performance in the research and possibly commercial environments as it will have a higher number of internal pre-configured loggers which can be started and stopped on the click of a button.

7.3.2 Analysis

The analysis presented in Chapter 5 provides the results gathered from the data collected as part of the longitudinal log study (Chapter 4). There are many interactions (particularly cross-application ones) that have not yet been studied. For example, cross-tab application in web browsers and loading images/media into Microsoft Word documents from file. These types of interactions can be analysed and results presented to users in many useful ways, through the provision of tools/visualisations which are applicable to these areas. For example, cross-tab interaction in web browsers could be aided by re-arranging relevant tabs to be positioned next to each other, making switching between them easier for users. These types of analyses would be valuable for interaction and interface designers as it would better inform the design of future iterations of programs and operating systems which can in turn better provide for the users' needs.

7.3.3 QuickFileAccess

The QuickFileAccess system reported in this thesis is in its infancy. There are many options for enhancing and improving these types of tools for the research community and general public. The QuickFileAccess tool supports all files/folders navigated to in Windows Explorer, displaying relevant items in the Windows Explorer Quick Access list. The algorithm used to present suggested folders to users relies on information on clipboard and file directory access interactions the user performs. Clipboard operations

are performed in multiple types of applications and thus generalises to all applications and interactions performed across the operating system as a whole. Systems are slowly moving online with the introduction of Microsoft Office online and online cloud-based file storage solutions. As a result of this the QuickFileAccess tool generalises well to other environments other than those it was originally designed to serve in that a version of it could be used within online file directory systems or online office environments (such as Dropbox or Microsoft Office online) in order to provide the same feedback to users as it does within Microsoft Windows desktop environments.

7.4 Cross-Application Interaction into the Future

The ways in which users interact with and utilise computers are unlikely to change as they have remained constant over a long period of time. It would be reasonable to assume that the use of cross-application interaction will continue to increase. However, users are choosing to use more online/cloud based services including Microsoft Office Online and Google Drive. These suites of applications allow online access to Microsoft Office Word, Excel, PowerPoint etc. and Google Docs, Sheets and Slides. These applications are accessed through a single web-browser and so window switching is less likely. A question of interest to the research community in this field is whether this will increase the amount of tab switching within web browsers and how will users cope with a higher number of switches if all interactions in the future occur in a single web browser window? The ways in which interaction between these online applications occurs and how users transfer data between each one is of relevance to this research when looking into the future. Interaction with applications is also becoming more mobile with more users than ever before interacting with various applications on mobile devices. Cross-application interaction in the area of mobile devices is currently poorly understood and requires further analysis. Mobile devices pose different issues because of smaller screen space and a more difficult ability to switch/transfer data between applications.

7.5 Limitations

Although the longitudinal log study undertaken in Chapter 4 gained a valuable insight into users' interactions on desktop operating systems, there are various limitations to

the research which, if addressed, could further increase the validity of the research and further benefit the research community as a whole.

7.5.1 Operating System

During the longitudinal log study only actions performed inside Microsoft Windows operating systems were logged. This limits the scope of the study as further valuable insights could be gained from studying other types of operating systems such as iOS or mobile platforms.

7.5.2 Participant Makeup

Within the log study only a small number of participants (17) were used limiting the better understanding which would have been achieved with a larger number of users. Valuable information such as whether users from different types of jobs or backgrounds interact differently would be an important addition to research in this area. Participants were recruited from family, friends and through connections within the School of Computing & Communications at Lancaster University. Various e-mails requesting assistance with the study were sent out to the department and once participants registered their interest in the study a participation information pack was sent which explained the study in detail and what they needed to do. 17 computer users from a variety of different backgrounds (11 frequent - more than 50 interaction days - and 5 female) took part in the user study for a period of 90 days. The age range of participants was between 18-74 years. Seven users had single screens, eight had two screens and two had three screens. Participants used a variety of Microsoft Windows operating systems with eight using Windows 7, three using Windows 8.1 and three using Windows 10. Seven users had laptops and eleven desktops.

As such, there is a certain bias which is a limitation of the study in that a lot of the participant cohort were work/research users and were found to generally use more programming IDEs/research tools/specialist programs, use more screens and usually be expert users. For each of the two main areas of analysis in Chapter 5 (window switching and data transfer), a brief analysis is provided in the following sentences on the participant cohort. Analysis conducted on the participant cohort for window switching,

clipboard and drag-and-drop showed actions were much more frequent among work users, possibly down to the fact they use their computers more frequently (of the total interaction hours recorded, 56.9% were among work users with 43.0% among leisure users). For window switching, over 80% of switches recorded were for work users, with less than 20% for leisure users). For clipboard usage these results were even higher for work users with over 90% of actions undertaken by work users and only around 9.5% by leisure users, indicating a higher usage rate of the clipboard among work users. The results for drag-and-drop followed a similar trend in that over 98% of drag-and-drop actions were performed by work users with just less than 2% of actions performed by leisure users. This data shows that the majority of window switches were for work users (80%) and work users utilised the clipboard and drag-and-drop operations much more frequently. These results were expected as it was observed that work users had a higher number of interaction hours and as such, it is to be expected work users will perform more window switches, clipboard and drag-and-drop actions than leisure users. As a result, this shows that it would be advantageous to utilise wider cohorts or various different cohorts in the future. It also shows that cohort-specific analysis is useful in understanding actions users undertake, due to the large variation in differences between work and leisure users.

7.5.3 Participant Makeup Reflection

The participant makeup could be seen to bias the results due to the fact the majority of participants came from the research community and from the School of Computing & Communications at Lancaster University. These users will generally be performing similar tasks (preparing lectures, writing research papers, performing research projects) and as such, these tasks could have had an impact on the results of the longitudinal log study. As such, the frequent use of some applications may be down to the fact these users are utilising the same applications to perform similar tasks (research/work). For example, high usage was seen in Mozilla Thunderbird, STEP 7 Manager and Rhino, all of which were from work users and higher than the usage of Google Chrome which was utilised by many more users. Of the top 10 used applications, 7 were directly related to a given work task while 3 (Windows Explorer , Google Chrome and Microsoft Outlook)

were less niche applications indicating high usage by individual users for an extended period of time.

A wider variety of participants would have been a useful addition to the study and would have given a better and more accurate overview of the ways in which users of computers perform interactions on them. Further deeper analysis into the different types of users would also have been beneficial and will be considered in any future work undertaken. Analysis by work vs home users and types of job performed for work users would have provided extra contextual information and may have led to being able to make more relevant statements surrounding these types of users and develop better tuned tools geared toward these users specifically.

Future studies undertaken will carefully take into account the participant makeup and extra work will be undertaken both in the participant recruitment stage and during the data analysis stage to fully understand the impact of the participant cohort selection. For example, participant recruitment will be expanded and participants from different backgrounds/workplaces will be recruited for future studies. Advertising regarding the longitudinal log studies will be more widely spread both in terms of across further departments across the university but also in external organisations/families. During the analysis stage of future studies, analysis by job or user type will be presented and compared to past results and with each other. This in turn will give a wider analysis taking into account the makeup of the participant cohort. It will also provide findings which are possibly relevant to different workers undertaking different jobs. Further analysing these interactions by type of participant may uncover different ways of working which require different means of assistance in order to increase efficiency. For example, work users may prefer and be more efficient with a different recommender tool (similar to QuickFileAccess) than home users. Extra analysis of the data would provide this valuable information. In summary, future studies will carefully take into account the participant cohort and analyse it fully.

7.5.4 Contextual Information

The study also had a limitation by the fact that no further contextual information was gathered. For example, no information relating to certain tasks the user was performing was gathered nor was additional information surrounding the rationale behind certain

user actions. Both these types of data would be valuable in gaining a deeper understanding of user interactions on desktop operating systems.

7.5.5 Data in Isolation

The analysis of the study described in Chapter 5 also only analysed data in isolation. For every participant, actions were only logged across one machine. Several of the participants of the study had multiple machines (for example an office computer (which interactions were logged on) and a laptop (which interactions were not logged on)). Logging all users' machines would be the ideal solution to this limitation but this is not currently possible if they have machines running different operating systems with which the software is not compatible. It would also allow comparisons to be made as to whether users interact in a cross-application way differently across different machines.

7.5.6 Contextual Information

The QuickFileAccess system makes the assumption that the relationship between clipboard and file directory access is a good correlation measure for reporting back relevant directories to the user through the Windows Explorer Quick Access system. There are also many other algorithms available for use in selecting which file directories to report back to the user through the Windows Explorer Quick Access list.

In summary, various limitations exist with the research which include only logging events within Microsoft Windows operating systems, using a small number of participants, not gathering extra contextual information, analysing data in isolation and the assumption that the QuickFileAccess system should use the relationship between clipboard and file directory access as a good measure.

7.6 Future Work

Understanding cross-application interaction is still not a solved problem. This thesis has provided a deep understanding of the ways in which cross-application interaction occurs in desktop environments. This section makes suggestions on ideas for future work.

7.6.1 Better Understanding Interactions of Users

Our longitudinal log study provided analysis of users' window switching, data transfer and file directory access. A further study could investigate other types of interactions not covered by this thesis such as tab switching in web browsers and/or within web applications and compare the ways in which tab switching occurs compared to how window switching happens. Further studies could also cover different types of operating systems, for example window switching on Apple or Linux machines. As interfaces and operating systems evolve it is expected that interaction methods in the area of cross-application may also change, especially with the switch to more web-based systems such as Microsoft Office and cloud-based file storage solutions. Studying these changes is of interest to interaction designers, operating system developers and application designers in general.

Researchers may also consider finding or coding new logging applications that can be easily plugged-in to the MultiLog logging framework to gain a richer understanding in other types of interaction methods such as the use of keyboard shortcuts or the ways in which interaction occurs with web browsers, even in the broader area of HCI in general. This data could be analysed in isolation or in conjunction with the existing logging applications that were used in this research plugged-in to MultiLog. Researchers are easily able to run the MultiLog software with these existing loggers and can simply add more to cater for their exact data collection requirements. Longer studies could be run in order to detect changes of trends in user interactions over a longer period of time (allowing the comparison of changing operating systems, jobs and research/work environments) or possibly the MultiLog framework could be modified to run on mobile devices or cloud-based services within web browsers to study the ways in which cross-application interaction occurs in these environments.

There were certain aspects of user interaction which cannot be fully understood through logging. The context around the ways in which interactions are performed cannot be understood through data logging alone and neither can the rationale behind why and how interactions are performed. Interviews or a direct observation method can help gain a better understanding of the context and rationale around user actions and can further increase the accuracy of results.

7.6.2 Development of Further Useful Tools/Visualisations

If researchers carry out further research in these areas using logging as a method of data collection the analysis of the results could help inform the design of new tools, similar to QuickFileAccess which can ultimately increase user efficiency by presenting more relevant data in the interfaces available. The QuickFileAccess tool could be improved to provide more in-depth algorithms of analysis to suggest the folders presented dynamically to the user through the tool. These algorithms could be fine-tuned to analyse the working patterns of certain tasks which could then display relevant folders based on these tasks. Secondly, the QuickFileAccess tool could be further extended to interface more with existing applications (for example, the default directories displayed on Open File or Save File dialog windows). If a user has, for example, been inserting lots of pictures from a directory into a Word document the tool, on opening, could select the relevant default directory where the pictures are located. There are many ways in which the tool could be extended to better assist users while they are performing tasks on their machines. This feedback to users could ultimately be used by future interface designers to cater better for users' needs. Further applications such as QuickFileAccess could be focussed on certain applications such as dynamically updating "suggested" lists based on the real-time analysis of data collected (either with MultiLog or other systems). This data could also possibly be used in the wider development of future operating systems by dynamically embedding "suggested" lists into the future design of such operating systems. For example, a list of "suggested websites" could be maintained within a web browser or "suggested tasks" based on to do lists or calendar entries. Another possible idea for the extension of tools such as QuickFileAccess is a system where users are able to configure which folders are displayed based on several different algorithms of analysing live data. For example, some users may wish the last ten file directories accessed to be listed, whereas others may benefit from the last two from Microsoft Word, Microsoft Excel and Microsoft PowerPoint to be listed in separate headings so that distinguishing between tasks is easier.

Chapter 8: Conclusion

This research was undertaken with an overarching goal to improve the efficiency of users when they are performing cross-application tasks. In order to do this an understanding of how certain interactions are performed had to be formed. In an attempt to achieve this, the research firstly understood how users were performing these interactions by developing the universal logging framework, MultiLog. Secondly, analysis from the data collected during a 90 day longitudinal log study was presented with various new and updated trends becoming apparent. Thirdly, an extension to the MultiLog logging framework was produced which analysed the collected data and presented relevant file directories back to the user through the Windows Explorer Quick Access menu. Results of the pilot study were also reported in the forms of both quantitative and qualitative data.

This thesis has made five primary contributions to the research knowledge in the domain of cross-application interaction. These are:

- 1 A review of related work in the areas of window switching, data transfer and cross-application interaction.
- 2 The development of MultiLog, a logging framework for merging multiple loggers' log files into one universal structure. Before this work it is not believed there was a tool available to researchers which would present the results of multiple log files in one repository in real time. The logging framework provides an easy approach to adding any pre-existing logging application to it via the in-built wizard interface or through the development of the configuration file (for deployments).

- 3 A 90 day longitudinal log study to collect data on user interactions/behaviour. Using the MultiLog logging framework, 17 participants' interactions were logged over a period of three months. This length of study not only allowed relevant results in the areas of window switching, data transfer and files to be presented but also allowed analysis and presentation of trends over time.

- 4 A characterisation of various types of interactions users perform on their machines. This thesis presented results in the primary areas of: (1) window switching, (2) data transfer and (3) file analysis. A deep analysis of the usage habits of all three types of interaction was performed resulting in a good overview of these types of interaction by users. Within (1) the methods of switching, position and size of windows, number and state at switch time were presented and discussed. The results identify new trends in window re-use and the frequency of switches. The window re-use and switching frequency rates have both increased (window re-use by 38% and switching frequency by 33 seconds compared to previous studies) meaning the same windows are being switched to on average 82% of the time. Most other previous trends were validated in the results of the study, however differences with window re-use were found. New characterisations were presented surrounding the on-screen placement of different types of windows, maximised vs. non-maximised windows, application-specific switching rates and the number of applications and windows open on single and multi-screen computers. Within (2) a deep understanding of how data transfer tools such as the Windows clipboard and drag-and-drop actions was presented and discussed. Findings showed that within application data transfer operations are more frequent than between application data transfer operations. A short time and low number of window switches between the copy/cut and paste operations was also observed. Similarly, within (3) an overview of the types of directories/files was presented surrounding the relationship between window switching and directories/files and also clipboard and directories/files. 23% of all Windows Explorer actions logged a directory at five levels deep in the file system. The next two most frequent directory levels logged were at levels six and four. The deepest level logged was level 14 while only 6% of actions were one level deep. 58% of actions occurred on drive C:\ while D:\ and H:\ were the next most used drive

letters and frequent usage of the same directories was observed across participants with some directories being used in excess of 600 times.

- 5 The enhancement of the MultiLog logging framework by the addition of QuickFileAccess. This system analyses the data collected by the MultiLog framework in real time and presents results to the user in the area of files/directories by dynamically modifying the Windows Explorer Quick Access list with files flagged as relevant by QuickFileAccess's software. While quantitative data showed a limited amount of interaction with the Windows Explorer Quick Access list the qualitative feedback from users indicated that the tool had improved their use of the Quick Access list and also helped them locate files quicker than before, in turn reducing the inefficient time they spend trying to find said files.

The Discussion chapter provides information regarding the limitations of the research, for example, the system only logging events within Microsoft Windows operating systems, using a small number of participants, not gathering extra contextual information, analysing data in isolation and the assumption that the QuickFileAccess system should use the relationship between clipboard and file directory access as a good measure. It also provides an overview of possible directions of future work in the area of cross-application interaction including but not limited to the better understanding of user interactions and the development of more tools which provide relevant feedback to the user to enhance their cross-application experience.

Researchers can use the MultiLog software to collect data from multiple sources to develop new interaction techniques. The QuickFileAccess software can be used as a starting point for the future tools and/or Windows Explorer plug-ins discussed in this thesis. Interaction designers may use these findings to inform the design of the next generation of cross-application interaction methods in modern operating systems.

References

- Abela, C., Staff, C., & Handschuh, S. (2010). *Task-based user modelling for knowledge work support*. Paper presented at the International Conference on User Modeling, Adaptation, and Personalization.
- Actual Keylogger. (2014). Actual Keylogger Software for Windows - Free Download. Retrieved from <http://www.actualkeylogger.com/>
- Adobe. Adobe ARM Logging. Retrieved from <http://www.adobe.com/devnet-docs/acrobatetk/tools/PrefRef/Windows/Updater-Win.html>
- Adobe Customer Improvement Program. (2014). Adobe Customer Improvement Program. Retrieved from <http://www.adobe.com/limited/apip.html>
- Adomavicius, G., & Tuzhilin, A. (2011). Context-aware recommender systems. In *Recommender systems handbook* (pp. 217-253): Springer.
- Alexander, J., Cockburn, A., Fitchett, S., Gutwin, C., & Greenberg, S. (2009). *Revisiting read wear: analysis, design, and evaluation of a footprints scrollbar*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Boston, MA, USA. <http://dx.doi.org/10.1145/1518701.1518957>
- Alexander, J., Cockburn, A., & Lobb, R. (2008). AppMonitor: A tool for recording user actions in unmodified Windows applications. *Behavior Research Methods*, 40(2), 413-421. doi:10.3758/BRM.40.2.413
- Bardram, J., Bunde-Pedersen, J., & Soegaard, M. (2006). *Support for activity-based computing in a personal computing operating system*. Paper presented at the Proceedings of the SIGCHI conference on Human Factors in computing systems.
- Bi, X., & Balakrishnan, R. (2009). *Comparing usage of a large high-resolution display to single or dual desktop displays for daily work*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Boston, MA, USA.
- Bihler, P., & Kniesel, G. (2007). *Seamless Cross-Application Workflow Support by User Interface Fusion*. Paper presented at the Multiple and Ubiquitous Interaction, Aarhus, Denmark.
- Biswas, P., & Robinson, P. (2010). *A brief survey on user modelling in HCI*. Paper presented at the Proc. of the International Conference on Intelligent Human Computer Interaction (IHCI).
- Brewster, S. A. (1998). *Sonically-enhanced drag and drop*. Paper presented at the International Conference on Auditory Display'98.
- Budzik, J., Hammond, K. J., & Birnbaum, L. (2001). Information access in context.
- CamStudio. (2014). CamStudio - Free Screen Recording Software. Retrieved from <http://camstudio.org/>
- Chapuis, O., & Roussel, N. (2007). *Copy-and-paste between overlapping windows*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, San Jose, California, USA.
- Chen, C., Perrault, S. T., Zhao, S., & Ooi, W. T. (2014). *BezelCopy: an efficient cross-application copy-paste technique for touchscreen smartphones*. Paper presented at the Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces, Como, Italy.
- Clemmensen, T. (2004). Four approaches to user modelling—a qualitative research interview study of HCI professionals' practice. *Interacting with computers*, 16(4), 799-829.

- Cosley, D., Lam, S. K., Albert, I., Konstan, J. A., & Riedl, J. (2003). *Is seeing believing? How recommender system interfaces affect users' opinions*. Paper presented at the Proceedings of the SIGCHI conference on Human factors in computing systems.
- Czerwinski, M., Smith, G., Regan, T., Meyers, B., Robertson, G., & Starkweather, G. (2003). *Toward Characterizing the Productivity Benefits of Very Large Displays*.
- Dostál, M., & Eichler, Z. (2011). *A research framework for performing user studies and rapid prototyping of intelligent user interfaces under the OpenOffice.org suite*. Paper presented at the Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems, Pisa, Italy. <http://dx.doi.org/10.1145/1996461.1996511>
- Dragunov, A. N., Dietterich, T. G., Johnsrude, K., McLaughlin, M., Li, L., & Herlocker, J. L. (2005). *TaskTracer: a desktop environment to support multi-tasking knowledge workers*. Paper presented at the Proceedings of the 10th international conference on Intelligent user interfaces, San Diego, California, USA.
- Drop, D. a. (2007). Drag and Drop. In *The Essential Guide to Flex 2 with ActionScript 3.0* (pp. 369-386). Berkeley, CA: Apress.
- Eklundh, K. S., & Kollberg, P. (2003). Emerging discourse structure: computer-assisted episode analysis as a window to global revision in university students' writing. *Journal of Pragmatics*, 35(6), 869-891. doi:10.1016/S0378-2166(02)00123-6
- Ezvid. (2014). Screen Recorder >> ezvid.com. Retrieved from <http://www.ezvid.com/screen-recorder>
- Faure, G., Chapuis, O., & Roussel, N. (2009). *Power tools for copying and moving: useful stuff for your desktop*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Boston, MA, USA.
- Fenstermacher, K. D., & Ginsburg, M. (2002). A Lightweight Framework for Cross-Application User Monitoring, 35(3), 51-59. Retrieved from <http://doi.ieeecomputersociety.org/10.1109/2.989930>
- Fischer, G. (2001). User modeling in human-computer interaction. *User Modeling and User-Adapted Interaction*, 11(1-2), 65-86.
- Fitchett, S., & Cockburn, A. (2012). *Accessrank: predicting what users will do next*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.
- Fitchett, S., Cockburn, A., & Gutwin, C. (2013). *Improving navigation-based file retrieval*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.
- Fono, D., & Vertegaal, R. (2005). *EyeWindows: evaluation of eye-controlled zooming windows for focus selection*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Portland, Oregon, USA.
- Gaylin, K. B. (1986). How are windows used? Some notes on creating an empirically-based windowing benchmark task. *SIGCHI Bull.*, 17(4), 96-100. doi:10.1145/22339.22355
- Google Chrome. (2014). Google Chrome Logging. Retrieved from <http://www.chromium.org/for-testers/enable-logging>
- Grabarnik, G., Salahshour, A., Subramanian, B., & Ma, S. (2004). *Generic adapter logging toolkit*. Paper presented at the Autonomic Computing, International Conference on.
- Grudin, J. (2001). *Partitioning digital worlds: focal and peripheral awareness in multiple monitor use*. Paper presented at the Proceedings of the SIGCHI

- Conference on Human Factors in Computing Systems, Seattle, Washington, USA.
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1), 10-18. doi:10.1145/1656274.1656278
- Hansen, G. (1997). Success in translation. *Perspectives: Studies in Technology*, 5(2), 10. Retrieved from <http://www.tandfonline.com/action/showCitFormats?doi=10.1080/0907676X.1997.9961310>
- Henderson, D. A. J., & Card, S. (1986). Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Trans. Graph.*, 5(3), 211-243. doi:10.1145/24054.24056
- Houben, S., Vermeulen, J., Luyten, K., & Coninx, K. (2012). *Co-activity manager: integrating activity-based collaboration into the desktop interface*. Paper presented at the Proceedings of the International Working Conference on Advanced Visual Interfaces, Capri Island, Italy.
- Hutchings, D. R., Smith, G., Meyers, B., Czerwinski, M., & Robertson, G. (2004). *Display space usage and window management operation comparisons between single monitor and multiple monitor users*. Paper presented at the Proceedings of the working conference on Advanced visual interfaces, Gallipoli, Italy. <http://dx.doi.org/10.1145/989863.989867>
- Hutchings, D. R., & Stasko, J. (2002). *QuickSpace: new operations for the desktop metaphor*. Paper presented at the CHI '02 Extended Abstracts on Human Factors in Computing Systems, Minneapolis, Minnesota, USA.
- Internet Explorer. Internet Explorer brndlog.txt Log File. Retrieved from <http://blogs.msdn.com/b/askie/archive/2012/06/20/internet-explorer-maintenance-brndlog-txt-what-is-it-and-how-to-use-it-when-troubleshooting.aspx>
- Jannach, D., Zanker, M., Ge, M., & Gröning, M. (2012). *Recommender systems in computer science and information systems—a landscape of research*. Paper presented at the International Conference on Electronic Commerce and Web Technologies.
- Jia, J., Sun, J., Tang, C.-K., & Shum, H.-Y. (2006). *Drag-and-drop pasting*. Paper presented at the ACM SIGGRAPH 2006 Papers, Boston, Massachusetts.
- Juvina, I., & Oostendorp, H. v. (2004). *Predicting user preferences: from semantic to pragmatic metrics of Web navigation behavior*. Paper presented at the Proceedings of the conference on Dutch directions in HCI, Amsterdam, The Netherlands. <http://dx.doi.org/10.1145/1005220.1005233>
- Kamba, T., Elson, S. A., Harpold, T., Stamper, T., & Sukaviriya, P. (1996). *Using small screen space more efficiently*. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Vancouver, British Columbia, Canada.
- Kang, Y.-a., & Stasko, J. (2008). *Lightweight task/application performance using single versus multiple monitors: a comparative study*. Paper presented at the Proceedings of Graphics Interface 2008, Windsor, Ontario, Canada.
- Kelly, D., & Sugimoto, C. R. (2013). A systematic review of interactive information retrieval evaluation studies, 1967–2006. *Journal of the American Society for Information Science and Technology*, 64(4), 745-770. doi:10.1002/asi.22799
- Kidlogger. Kidlogger. Retrieved from <http://kidlogger.net/>

- Kinkead, R. (1975). Typing Speed, Keying Rates, and Optimal Keyboard Layouts. *Proceedings of the Human Factors and Ergonomics Society*, 19(2), 159-161. doi:10.1177/154193127501900203
- Kobayashi, M., & Igarashi, T. (2007). *Boomerang: suspendable drag-and-drop interactions based on a throw-and-catch metaphor*. Paper presented at the Proceedings of the 20th annual ACM symposium on User interface software and technology, Newport, Rhode Island, USA.
- Kukreja, U., Stevenson, W., & Ritter, F. (2006). RUI: Recording user input from interfaces under Windows and Mac OS X. *Behavior Research Methods*, 38(4), 656-659. doi:10.3758/BF03193898
- Kumar, M., Paepcke, A., & Winograd, T. (2007). EyeExposé: Switching applications with your eyes. In: Citeseer.
- Ladakis, E., Koromilas, L., Vasiliadis, G., Polychronakis, M., & Ioannidis, S. (2013). *You Can Type, but You Can't Hide: A Stealthy GPU-based Keylogger*. Paper presented at the EuroSec'13, Prague, Czech Republic. <http://www.cs.columbia.edu/~mikepo/papers/gpukeylogger.eurosec13.pdf>
- Leijten, M., & Waes, L. V. (2005). *Inputlog: A logging tool for the research of writing processes*. Paper presented at the Working Papers 2005011, University of Antwerp, Faculty of Applied Economics. http://www.researchgate.net/publication/4983063_Inputlog_A_logging_tool_for_the_research_of_writing_processes/file/60b7d519ce23bcc0db.pdf
- Lieberman, H. (1995). Letizia: An agent that assists web browsing. *IJCAI (1)*, 1995, 924-929.
- Linton, F., Joy, D., & Schaefer, H.-P. (1999). Building user and expert models by long-term observation of application usage. In *UM99 User Modeling* (pp. 129-138): Springer.
- Log Monitor. (2014). Log Monitor - CodeProject. Retrieved from <http://www.codeproject.com/Articles/19062/Log-Monitor>
- Logisoft. Revealer Keylogger. Retrieved from <http://www.softactivity.com/spy-software.asp>
- Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook* (pp. 73-105): Springer.
- Mackinlay, J. D., & Royer, C. (2004). Log-based longitudinal study finds window thrashing. *Palo Alto Research Center*.
- Math, M. M., Seetha, M., Kulkarni, U. P., & Yardi, A. R. (2009). Generic Log Adapters- A Step towards building a Parser Based Self Healable Autonomic System. *International Journal of Recent Trends in Engineering*, 2(3), 102-107.
- May, J., Barnard, P. J., & Blandford, A. (1993). Using structural descriptions of interfaces to automate the modelling of user cognition. *User Modeling and User-Adapted Interaction*, 3(1), 27-64.
- Mendeley. Mendeley Logging. Retrieved from <http://support.mendeley.com/customer/portal/articles/227943-how-do-i-locate-mendeley-desktop-s-log-files->
- Microsoft Corporation. Microsoft Problem Steps Recorder. Retrieved from <http://windows.microsoft.com/en-gb/windows7/how-do-i-use-problem-steps-recorder>
- Microsoft Corporation. (2003). What is the enable logging (troubleshooting) option? - Outlook. Retrieved from <http://office.microsoft.com/en-gb/outlook-help/what-is-the-enable-logging-troubleshooting-option-HA001174266.aspx>

- Microsoft Customer Experience Improvement Program. (2009). Microsoft Customer Experience Improvement Program. Retrieved from <http://www.microsoft.com/products/ceip/en-US/default.aspx>
- Microsoft Distributed Transaction Coordinator. Microsoft Distributed Transaction Coordinator Log File. Retrieved from <http://technet.microsoft.com/en-us/library/cc759136%28v=ws.10%29.aspx>
- Microsoft Outlook. (2014). Microsoft Outlook Logging. Retrieved from <http://office.microsoft.com/en-gb/outlook-help/what-is-the-enable-logging-troubleshooting-option-HA001230421.aspx>
- Microsoft Research. PersonalVibe. Retrieved from <http://research.microsoft.com/en-us/downloads/0ea12e49-8b29-4930-b380-a5a00872d229/>
- Microsoft Visual Studio. (2014). /Log (devenv.exe). Retrieved from <http://msdn.microsoft.com/en-us//library/ms241272.aspx>
- Miryung, K., Bergman, L., Lau, T., & Notkin, D. (2004, 19-20 Aug. 2004). *An ethnographic study of copy and paste programming practices in OOPL*. Paper presented at the Empirical Software Engineering, 2004. ISESE '04. Proceedings. 2004 International Symposium on.
- Montgomery, A., & Faloutsos, C. (2001). Identifying Web Browsing Trends and Patterns *Internet Watch*. Retrieved from <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=933515&tag=1>
- Mozilla Firefox. Mozilla Firefox Logging. Retrieved from https://developer.mozilla.org/en-US/docs/Mozilla/Debugging/HTTP_logging
- Mozilla Thunderbird. Mozilla Thunderbird Logging. Retrieved from <https://wiki.mozilla.org/MailNews:Logging>
- Oliver, N., Smith, G., Thakkar, C., & Surendran, A. C. (2006). *SWISH: semantic analysis of window titles and switching history*. Paper presented at the Proceedings of the 11th international conference on Intelligent user interfaces, Sydney, Australia. <http://dx.doi.org/10.1145/1111449.1111492>
- Peres, S. C., Tamborello, F. P. I., Fleetwood, M. D. I., Chung, P. I., & Paige-Smith, D. L. I. (2004). Keyboard Shortcut Usage: The Roles of Social Factors and Computer Experience. *Proceedings of the Human Factors and Ergonomics Society*, 48(5), 803-807. doi: 10.1177/154193120404800513
- Perugini, S., Gonçalves, M. A., & Fox, E. A. (2004). Recommender systems research: A connection-centric survey. *Journal of Intelligent Information Systems*, 23(2), 107-143.
- REFOG. (2014). REFOG Free Keylogger. Retrieved from <http://www.refog.com/free-keylogger.html>
- Rylstim Screen Recorder. (2014). Rylstim Screen Recorder - Reocrding On-Screen Manipulations to AVI File | Sketchman Studio. Retrieved from <http://www.sketchman-studio.com/rylstim-screen-recorder/>
- Shahzad, K. (2013). *Generic Log Analyser*. (Master of Science). Lulea University of Technology, Retrieved from <https://pure.ltu.se/portal/files/42129705/LTU-EX-2013-42114123.pdf>
- Shih, C.-H., Huang, H.-C., Liao, Y.-K., Shih, C.-T., & Chiang, M.-S. (2010). 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(2), 416-425. doi:<http://dx.doi.org/10.1016/j.ridd.2009.10.004>

- Shneiderman, B., & Kang, H. (2000, 2000). *Direct annotation: a drag-and-drop strategy for labeling photos*. Paper presented at the Information Visualization, 2000. Proceedings. IEEE International Conference on.
- Skype. Skype. Retrieved from <http://www.skype.com/>
- Soft Activity. SoftActivity Keylogger. Retrieved from <http://www.softactivity.com/spy-software.asp>
- Stolee, K. T., Elbaum, S., & Rothermel, G. (2009, 20-24 Sept. 2009). *Revealing the copy and paste habits of end users*. Paper presented at the 2009 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC).
- Sunderaraman, P. (2013). Drag and Drop. In *Practical Ext JS 4* (pp. 131-152). Berkeley, CA: Apress.
- Tak, S. (2011). *Understanding and Supporting Window Switching*. University of Canterbury,
- Tak, S., & Cockburn, A. (2009). *Window Watcher: a visualisation tool for understanding windowing activities*. Paper presented at the Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7, Melbourne, Australia. http://link.springer.com/chapter/10.1007/978-3-642-03658-3_25
- Tak, S., Cockburn, A., Humm, K., Ahlström, D., Gutwin, C., & Scarr, J. (2009). Improving Window Switching Interfaces. In T. Gross, J. Gulliksen, P. Kotzé, L. Oestreicher, P. Palanque, R. Prates, & M. Winckler (Eds.), *Human-Computer Interaction – INTERACT 2009* (Vol. 5727, pp. 187-200): Springer Berlin Heidelberg.
- Tan, D. S., Meyers, B., & Czerwinski, M. (2004). *WinCuts: manipulating arbitrary window regions for more effective use of screen space*. Paper presented at the CHI '04 Extended Abstracts on Human Factors in Computing Systems, Vienna, Austria.
- TeamViewer. (2014). TeamViewer - Free Remote Control, Remote Access & Online Meetings. Retrieved from <http://www.teamviewer.com/>
- TechSmith. (2015). Morae. Retrieved from <https://www.techsmith.com/morae.html>
- Tiny Log Monitor. (2014). Tiny Log Monitor. Retrieved from <http://www.tinylogmonitor.com/>
- Ubuntu. LinuxLogFiles - Community Help Wiki. Retrieved from <https://help.ubuntu.com/community/LinuxLogFiles>
- University of Washington. WebQuilt. Retrieved from <http://dub.washington.edu:2007/pubs/#webquilt>
- User Logger. User Logger. Retrieved from <http://chemware.co.nz/usrlog.htm>
- VMWare. VMWare Log Files. Retrieved from http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1021806
- Voida, S., Mynatt, E. D., & Edwards, W. K. (2007). *Giornata: Re-Envisioning the Desktop Metaphor to Support Activities in Knowledge Work*. Paper presented at the ACM User Interface Software and Technology Symposium, Newport, Rhode Island, USA.
- Windows Update. Windows Update Log Files. Retrieved from <http://support.microsoft.com/kb/902093>
- WindowsTechies. (2019). Exploring the Quick Access Section in Windows 10's File Explorer. Retrieved from <https://windowstechies.com/exploring-the-quick-access-section-in-windows-10s-file-explorer/>
- Wintective. (2014). Wintective. Retrieved from <http://www.oocities.org/wintective/>

Appendices

Appendix 1: Study: Information for Participants

This appendix is the “Information for Participants” pack which was sent out when interest in the study was shown from potential participants. If they were interested in participating, they were asked to review the documents, sign the consent page and return it to the researcher.

Understanding how users employ multiple desktop applications together in order to complete tasks



Jonathan Woodruff | j.woodruff@lancaster.ac.uk

Invitation for Study Participation

Dear Participant,

The aim of our research is to gain a deeper understanding of how users of computers interact with different applications while completing tasks, such as writing a paper or processing e-mails. We invite you to participate in our study which uses logging software to monitor many internal systems and applications installed on your computer to collect information about actions you perform whilst using it. The software is easily installed by you and is fully-configurable to allow you to pause all logging and delete individual log entries. The data is automatically uploaded to our server every hour and so nothing other than installation is required by you. The software runs on Windows 7, Windows 8, Windows 8.1 and Windows 10 and you will be asked to run the software for a period of 3 months. If you are interested in participating in our study or require further information, please contact Jonathan Woodruff on j.woodruff@lancaster.ac.uk where a Participation Information Sheet and Consent Form and/or additional details will be provided. You can also view more details on the website at: www.scc.lancs.ac.uk/MultiLog/. A sample log is included on the website.

Kind regards,

Jonathan Woodruff & Jason Alexander
School of Computing & Communications

Jonathan Woodruff | j.woodruff@lancaster.ac.uk

Participation Information Sheet

What is this study about?

The purpose of this study is to find out how users of computers utilise applications in a task-based way. We are looking at how users interact with different applications on their machines in order to complete certain tasks such as “processing e-mails”, “writing a report” or “writing a presentation”. Most of these tasks require the use of more than one application and this study is about the interactions between these applications while performing a certain task.

Why have I been approached?

You have been approached because the study requires information from people who utilise computers on a regular basis to complete sets of tasks. We are in need of twenty users of Windows 7, Windows 8, Windows 8.1 and Windows 10 in order to obtain good quality data hence the approach.

Do I have to take part?

No. Participation is voluntary and apart from a short application installation, time from you is minimal (the software runs automatically after first install so as to be as less time-consuming as possible). A small interview may take place before and after the study to allow us to ask qualitatively about your computer use habits and gather demographic data.

What will I be asked to do if I take part?

If you decide you would like to take part, you will be asked to complete a consent form, demographics documentation and take part in a small interview (after the study has concluded) in order for us to gather qualitative information regarding your computer use habits. Before the commencement of the study you will have the chance to liaise with the researchers to ensure all questions you may have are answered. On commencement of the study, you will be required to install a small piece of logging software on your computer, full details of how to do this will be provided. At all times you can pause logging or delete individual log entries if you do not wish for them to be included in the output. Examples of log data collected are:

- Mouse clicks (left and right) including what was clicked, in which application, and the type of click (single/double).
- Mouse drags including what was dragged and in which application.
- Mouse scroll bar interactions and where this was performed.
- Keyboard input including where keyboard input was directed.
- Window switches including the name of the window switched to and the process information (name and ID) of the application the window belongs to. The dimensions and position of the window is also recorded.
- Drag-and-drop actions including the name of the object (file, e-mail etc.) being dragged, the title of the window it was dragged from and the title of the window it was dragged to.

- Clipboard cut, copy and paste actions.
 - For text, only the length of the text is recorded, not the actual text.
 - For files, the names of the file(s) are recorded.
 - For audio and images, only “Audio” and “Images” are recorded.
 - For all actions, the window where the cut/copy/paste occurred will be recorded.
- Process information which includes the name, ID number and parent process of processes started on your computer by you or the system.

The logs are automatically uploaded to our secure server every hour and so no more input is required by you. At the end of the study instructions of how to remove the logging software will be provided unless you wish to be considered for future studies which may include the use of the same software.

Will my data be identifiable?

The data collected for this study will be stored securely and only the researchers conducting this study will have access to this data:

- Log data will be destroyed and/or deleted once the project has been submitted for publication/examined.
- Log data will be anonymised on our web server so that only the researchers know which participant the data is from. Data will be anonymised for inclusion in any publications arising from the work.
- Hard copies of interview transcripts will be kept in a locked cabinet.
- The web server holding the log data is secure and logs are downloaded to a University-approved secure storage area daily and removed from the web server to ensure extra security.
- Any typed versions of interviews conducted will be made anonymous by removing any identifying information including your name. Anonymised direct quotations from your interview may be used in the reports or publications from the study, so your name will not be attached to them.
- All your personal data will be confidential and will be kept separately from any interview responses.

There are some limits to confidentiality: if what is said in any interview or found in any log entry concerns the researchers that you or someone else is of significant risk of harm or if any illegal activities are discovered, the confidentiality rules above will have to be broken and a member of senior staff, or in severe cases, the police may have to be contacted.

What will happen to the results?

The results will be summarised and reported in a dissertation/thesis and may be submitted for publication in an academic or professional journal. Any identifying information will be removed before submission.

Are there any risks?

There are no risks anticipated with participating in this study. However, if you experience any concerns or distress following or during participation you are encouraged to inform the researcher as soon as possible.

Are there any benefits to taking part?

Although you may find participating interesting, there are no direct benefits to taking part. At the end of the study if interested, participants will be provided with a copy of the results/outcomes of the study and an opportunity to speak to the researcher will be available.

Who has reviewed this report?

The study has been reviewed by Jason Alexander (MPhil Supervisor) in the School of Computing & Communications and approved by the University Research Ethics Committee at Lancaster University.

Where can I obtain further information about the study if I need it?

If you have any questions about the study, please contact the main researcher:

Jonathan Woodruff

+44 (0)1524 510371

j.woodruff@lancaster.ac.uk

Complaints

If you wish to make a complaint or raise concerns about any aspect of this study and do not want to speak to the researcher, you can contact:

Professor Awais Rashid (Director of Research)

+44 (0)1524 510316

a.rashid@lancaster.ac.uk

Understanding how users employ multiple desktop applications together in order to complete tasks



Jonathan Woodruff | j.woodruff@lancaster.ac.uk

Consent Form

Participant Name:

Title of Study: Understanding how users employ multiple desktop applications together in order to complete tasks.

Researcher: Jonathan Woodruff & Jason Alexander
School of Computing & Communications
InfoLab21, Lancaster University, South Drive, Lancaster, LA1
4WA, UK.

j.woodruff@lancaster.ac.uk
+44 (0)1524 510371

Before agreeing to participate in this research study, we kindly ask you to read and understand the following explanation of the study and how it will be conducted.

Purpose of the Study

The purpose of the study is to gather data about how users of computers utilise different applications in a task-based way.

Procedures

This study involves the installation of logging software on your computer. After you have expressed your interest and signed up, the study will begin. You will be sent instructions (or be visited by the researcher if local) on how to install the logging software on your computer which will also include details of how to pause logging and delete individual log entries for privacy reasons. We will also ask you to complete a small demographic questionnaire. Once installation is complete, the software runs automatically for the duration of the study and data is automatically uploaded to our web server once hourly. At the end of the study there may be a brief questionnaire and full details of how to remove the software from your computer will be provided if you wish to do so. There may be an opportunity to participate in further studies, details will be provided at the end of the initial study if appropriate.

Confidentiality

Information collected during the study will be confidential. All identities will be anonymised and no names will be revealed to anyone other than the researchers. The anonymised data may be included in scientific publications. Log data uploaded to the web server automatically is downloaded and stored privately on University-approved secure storage systems with the web server copy being destroyed for extra security.

Risks

There are no risks to the participants during the study.

Further Information

To contact an independent person about this research, please refer to:
Professor Awais Rashid (Director of Research)
School of Computing & Communications
D41, InfoLab21, Lancaster University, South Drive, Lancaster, LA1 4WA,
UK.
+44 (0)1524 510316
a.rashid@lancaster.ac.uk

Consent

I have had the opportunity to discuss this study and my questions have been answered to my satisfaction. I consent to take part in the study with the understanding I may withdraw at any time. I voluntarily consent to participate in this study.

Signatures

Participant: _____

Date:

Investigator: _____

Date:

Appendix 2: Study: Installation Instructions

The Study Installation Instructions were provided once the consent form (Appendix 1) was reviewed, completed and returned. The instructions contain step-by-step details explaining how to install the MultiLog software.

Jonathan Woodruff | j.woodruff@lancaster.ac.uk

MultiLog Logging Software Information Sheet

Thank you for agreeing to use the MultiLog software for the duration of the study. This document is split into two sections: set up and use. If the researchers are performing the set up of the software with you, please ignore this section.

MultiLog Set Up

Please follow the following instructions to set up MultiLog manually without the assistance of the researchers:

1. Extract (unzip) the “MultiLog.zip” file supplied to any location on the computer’s file system.
2. Click “Start”, type “Task Scheduler” (or “Schedule Tasks” – Windows 8/8.1) and open the Task Scheduler.
3. In the left-hand panel, click “Task Scheduler Library”.
4. In the right-hand panel, click “Create Task”.
5. In the “Name” box, type “MultiLog”.
6. Check “Run with highest privileges”.
7. Ensure “Configure for” is set to “Windows Vista, Windows Server 2008”.
8. Click the “Triggers” tab. Click “New”, select “At logon” in the “Begin the task” dropdown box and click “Ok”.
9. Click the “Actions” tab. Click “New”, click “Browse”, browse to the location where you unzipped the files and select “MultiLog.exe”. Click “Ok”.
10. Click the “Conditions” tab. Un-tick “Start the task only if the computer is on AC power”.
11. Click “Ok” to close the “Create Task” dialogue.
12. Go to the folder where you un-zipped the contents of the zip file and start “MultiLog.exe”.
13. MultiLog will start minimised to the system tray.

Using MultiLog

MultiLog requires no additional input from the user to log effectively. There are however a few functions available to the user if they wish to use them:

1) The MultiLog Interface/viewing live log entries

To view live log entries, locate the “ML” MultiLog icon in the system tray. Right-click it and select “Show MultiLog”. The MultiLog interface will open and the most recent logs will be available for you to view. Logs are ordered by timestamp (1st column) and are classified in categories by the logger the produced the data (3rd column). The 2nd column is the data produced by the logger.

2) Delete an individual log line

To delete an individual log line, firstly ensure the MultiLog interface is open (see 1)). Locate the line you wish to delete. Right-click the line and click “Delete”. This will remove it from the interface and the MultiLog database.

3) Pause Logging

To pause logging at any time, firstly ensure the MultiLog interface is open (see 1)). Simply click “Pause Logging”. This will stop all logging in all loggers until the option is de-selected, which will resume logging.

Appendix 3: Study: Demographics Survey

The demographics survey was used to collect information about participants and was available in paper form and also online.

MultiLog Demographics Survey

1. Please tell us your name: *

2. Please indicate your gender: *

Male

Female

3. Please indicate your age: *

Under 18

18-24 years old

25-34 years old

35-44 years old

45-54 years old

55-64 years old

65-74 years old

75 years or older

4. Please indicate your employment status: *

Employed (salaried staff)

Self-employed

Company director/owner

PhD student

Undergraduate student

Unemployed

Other (please specify):

5. What is your highest level of education completed: *

- High school
- College (UK)
- University Bachelor's degree
- University Master's degree
- University Doctorate degree
- Other (please specify):

6. If you are related to a University, please state which one and your subject area:

7. Please indicate if you use your computer for any of the following WORK tasks:

- E-mail
- Word processing
- Spreadsheets
- Presentation preparation
- Web browsing
- Programming
- Academic paper writing
- University/school/homework
- Data analysis
- Other (please specify):

8. Please indicate if you use your computer for any of the following HOME tasks:

- E-mail

- Web browsing
- Online games
- Installed (local) games
- Household admin tasks (such as paying bills)
- Watching video content
- Listening to music
- Voluntary work for organisations
- Word processing
- Other (please specify):

9. If you use your computer for work, please tell us your job title:

10. The following questions concern the computer MultiLog is/will be installed on.
What type of computer is MultiLog to run on: *

- Desktop
- Laptop
- Tablet
- Other (please specify):

11. What operating system does this computer use: *

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10

12. Where is this computer usually situated: *

- Desktop on desk
- Laptop on desk
- Laptop on lap
- Laptop in mobile location (train, plane etc.)
- Other (please specify):

13. If you use a laptop, do you use the touchpad or a connected mouse:

- Touchpad
- Connected mouse

Appendix 4: QuickFileAccess Pilot Study: Information for Participants

This appendix is the “Information for Participants” pack which was sent out when interest in the QuickFileAccess pilot study was shown from potential participants. If they were interested in participating, they were asked to review the documents, sign the consent page and return it to the researcher.

Jonathan Woodruff | j.woodruff@lancaster.ac.uk

Invitation for Study Participation

All study participants will be recruited through e-mail using the following text. Departmental mailing lists and personal contacts will be used for recruitment. The following is the text that will be included:

Dear Participant,

The aim of our research is to gain a deeper understanding of how users of computers interact with different applications while completing tasks, such as writing a paper or processing e-mails. We invite you to participate in our study which uses logging software to monitor many internal systems and applications installed on your computer to collect information about actions you perform whilst using it. It also provides feedback analysing the actions you perform in order to make it easier for you to interact with your computer by providing visualisations and additional information such as relevant directories to save files in. The software is easily installed by you and is fully-configurable to allow you to turn on and off parts of logging from particular systems or applications, pause all logging or delete individual log entries. The software runs on Windows 10 and you will be asked to run the software for a period of 2-3 weeks. If you are interested in participating in our study or require further information, please contact Jonathan Woodruff on j.woodruff@lancaster.ac.uk where a Participation Information Sheet and Consent Form and/or additional details will be provided. You can also view more details on the website at: www.jonathanwoodruff.co.uk/MultiLog/. A sample log is included on the website.

Kind regards,

Jonathan Woodruff & Jason Alexander

School of Computing & Communications

Jonathan Woodruff | j.woodruff@lancaster.ac.uk

Participation Information Sheet

For further information about how Lancaster University processes personal data for research purposes and your data rights please visit our webpage: www.lancaster.ac.uk/research/data-protection.

What is this study about?

The purpose of this study is to find out how users of computers utilise applications in a task-based way and provide feedback to users in order to increase their efficiency (for example, providing visualisations showing recommended file directories). We are looking at how users interact with different applications on their machines in order to complete certain tasks such as “processing e-mails”, “writing a report” or “writing a presentation”. Most of these tasks require the use of more than one application and this study is about the interactions between these applications while performing a certain task and also includes feedback to users showing relevant file directories as these may be of use to the user.

Why have I been approached?

You have been approached because the study requires information from people who utilise computers on a regular basis to complete sets of tasks. We are in need of ten users of Windows 10 in order to obtain good quality data hence the approach.

Do I have to take part?

No. Participation is voluntary and apart from a short application installation, time from you is minimal (the software runs automatically after first install so as to be as less time-consuming as possible).

What will I be asked to do if I take part?

If you decide you would like to take part, you will be asked to complete a consent form and demographics documentation. The consent form will be hand delivered, posted to you or e-mailed to you. The consent form should be printed, signed with a pen and then returned to the researcher. It should be returned by either: (1) arranging collection by the researcher, (2) by posting the consent form back to the University or (3) sending a scanned signed copy of the form by e-mail to the researcher.

The demographics form can either be hand delivered, posted to you, e-mailed to you or filled in online at www.jonathanwoodruff.co.uk/multilog/survey/, depending on your preference. The demographics questionnaire should be returned either by: (1) arranging collection by the researcher, (2) by posting the consent form back to the University or (3) sending a scanned signed copy of the form by e-mail to the researcher. If you have filled the demographics questionnaire in online, you do not need to return a paper or scanned copy.

Before the commencement of the study you will have the chance to liaise with the researchers to ensure all questions you may have are answered. You will be sent instructions (or be visited by the researcher if local) of how to install the logging software on your computer which will also include details of how to start and stop logging, pause logging and delete individual log entries for privacy reasons. Once installation is complete, the software runs automatically for the duration of the study and data is automatically uploaded to our web server once daily. Examples of log data collected are different window switches, websites visited, e-mails sent and received including the sender/recipient but not the message contents, copy and paste and drag and drop actions. Low level inputs such as keystrokes are not recorded as part of this study. The logs are automatically uploaded to our server every day and so no more input is required by you. At the end of the study instructions of how to remove the logging software will be provided unless you wish to be considered for future studies which may include the use of the same software. At the end of the study full details of how to remove the software from your computer will be provided if you wish to do so. There may be an opportunity to participate in further studies, details will be provided at the end of the initial study if appropriate.

Will my data be identifiable?

The data collected for this study will be stored securely and only the researchers conducting this study will have access to this data:

- Log data will be destroyed and/or deleted once the project has been submitted for publication/examined.
- You have the option to hash (make unreadable) full website URLs from the logging system interface.
- The web server holding the log data is secure and logs are downloaded to a storage area daily and removed from the web server to ensure extra security.

There are some limits to confidentiality: if what is found in any log entry concerns the researchers that you or someone else is of significant risk of harm or if any illegal activities are discovered, confidentiality will have to be broken and a member of senior staff, or in severe cases, the police may have to be contacted.

What will happen to the results?

The results will be summarised and reported in a dissertation/thesis and may be submitted for publication in an academic or professional journal. Any identifying information will be removed before submission.

Can I withdraw from the study?

Yes, you can withdraw from the study at any time during the study or for a period of 2 weeks after its conclusion. After this period of time, withdrawal is not possible as data will have started to be analysed and possibly published in academic papers however all data will be anonymised so you will never publicly be identifiable. To withdraw, simply notify: j.woodruff@lancaster.ac.uk of your want to withdraw. All data will be deleted once your withdrawal request is received. In order to remove the software, please follow the step-by-step instructions in the debrief sheet. If you require any further assistance with withdrawal, please do not hesitate to contact: j.woodruff@lancaster.ac.uk.

Are there any risks?

There are no risks anticipated with participating in this study. However, if you experience any concerns or distress following or during participation you are encouraged to inform the researcher as soon as possible.

Are there any benefits to taking part?

Although you may find participating interesting, there are no direct benefits to taking part. At the end of the study if interested, participants will be provided with a copy of the results/outcomes of the study and an opportunity to speak to the researcher will be available.

Who has reviewed this report?

The study has been reviewed by Jason Alexander (MPhil Supervisor) in the School of Computing & Communications and approved by the FST Research Ethics Committee at Lancaster University.

Where can I obtain further information about the study if I need it?

If you have any questions about the study, please contact the main researcher:

Jonathan Woodruff

+44 (0)1524 510371

j.woodruff@lancaster.ac.uk

Complaints

If you wish to make a complaint or raise concerns about any aspect of this study and do not want to speak to the researcher, you can contact:

Professor Nicholas Race (Director of Research)

+44 (0)1524 510123

n.race@lancaster.ac.uk

Understanding how users employ multiple desktop applications together in order to complete tasks



Jonathan Woodruff | j.woodruff@lancaster.ac.uk

Consent Form

Participant Name: _____

Title of Study: Understanding how users employ multiple desktop applications together in order to complete tasks

Researcher: Jonathan Woodruff & Jason Alexander
School of Computing & Communications
InfoLab21, Lancaster University, South Drive, Lancaster, LA1
4WA, UK.

j.woodruff@lancaster.ac.uk

+44 (0)1524 510371

Withdrawal?

You can withdraw from the study at any time during the study or for a period of 2 weeks after its conclusion. After this period of time, withdrawal is not possible as data will have started to be analysed and possibly published in academic papers however all data will be anonymised so you will never publicly be identifiable. To withdraw, simply notify: j.woodruff@lancaster.ac.uk of your want to withdraw. All data will be deleted once your withdrawal request is received. In order to remove the software, please follow the step-by-step instructions in the debrief sheet. If you require any further assistance with withdrawal, please do not hesitate to contact:

j.woodruff@lancaster.ac.uk.

Further Information

To contact an independent person about this research, please refer to:

Professor Nicholas Race (Director of Research)

School of Computing & Communications

D33, InfoLab21, Lancaster University, South Drive, Lancaster, LA1 4WA, UK.

+44 (0)1524 510123

n.race@lancaster.ac.uk

Please tick each box. Please complete this form in pen.

1. I confirm that I have read and understand the information sheet for the above study. I have had the opportunity to consider the information, ask questions and have had these answered satisfactorily

2. I understand that my participation is voluntary and that I am free to withdraw at any time, without giving any reason. If I withdraw within two weeks of commencement of the study my data will be removed. If I am involved in focus groups and then withdraw my data will remain part of the study.

3. I understand that any information given by me may be used in future reports, academic articles, publications or presentations by the researcher/s, but my personal information will not be included and I will not be identifiable.

4. I understand that my name/my organisation's name will not appear in any reports, articles or presentation without my consent.

5. I understand that data will be kept according to University guidelines for a minimum of 10 years after the end of the study.

6. I agree to take part in the above study.

Consent

I have had the opportunity to discuss this study and my questions have been answered to my satisfaction. I consent to take part in the study with the understanding I may withdraw at any time. I voluntarily consent to participate in this study.

Signatures

Participant: _____

Date: _____

Investigator: _____

Date: _____

One copy of this form will be given to the participant and the original kept in the files of the researcher at Lancaster University.

Please return this signed completed form to the researcher in one of the following three ways:

1. Arrange collection of this form with the researcher.
2. Post this form to Jonathan Woodruff, Office D22, School of Computing & Communications, InfoLab21, Lancaster, LA1 4WA.
3. Scan this form in and send the signed scanned copy to j.woodruff@lancaster.ac.uk.

Appendix 5: QuickFileAccess Pilot Study: Installation Instructions

The QuickFileAccess Installation Instructions were provided once the consent form (Appendix 4) was reviewed, completed and returned. The instructions contain step-by-step details explaining how to install the QuickFileAccess software.

Jonathan Woodruff | j.woodruff@lancaster.ac.uk

MultiLog Quick Access Study Information Sheet

Thank you for agreeing to the MultiLog Quick Access Study.

Interface Feedback via the Windows Explorer Quick Access list

If you were involved in the first study we performed, you will know we collected data surrounding the use of applications on desktop operating system environments. This small study follows on and analyses the ways in which you open/save files and how copy/cut/paste operations are performed. It uses this collected information to manage the Windows Quick Access list within Windows Explorer to attempt to present more relevant folders to you to improve the ways in which you access folders/files on your computer. At the end of the study, we will ask questions surrounding your use of the Quick Access system and whether you found the results presented there by the MultiLog software useful.

MultiLog Quick Access Study Set Up

Please follow the following instructions to set up MultiLog manually without the assistance of the researchers:

1. Extract (unzip) the “MultiLog-QuickAccess.zip” file supplied to any location on the computer’s file system.
2. Click “Start”, type “Task Scheduler” (or “Schedule Tasks” – Windows 8/8.1) and open the Task Scheduler.
3. In the left-hand panel, click “Task Scheduler Library”.
4. In the right-hand panel, click “Create Task”.
5. In the “Name” box, type “MultiLog”.
6. Check “Run with highest privileges”.
7. Ensure “Configure for” is set to “Windows Vista, Windows Server 2008”.
8. Click the “Triggers” tab. Click “New”, select “At logon” in the “Begin the task” dropdown box and click “Ok”.
9. Click the “Actions” tab. Click “New”, click “Browse”, browse to the location where you unzipped the files and select “FinalProject.exe”. Click “Ok”.
10. Click the “Conditions” tab. Un-tick “Start the task only if the computer is on AC power”.
11. Click “Ok” to close the “Create Task” dialogue.
12. Go to the folder where you un-zipped the contents of the zip file and start “FinalProject.exe”.
13. MultiLog will start minimised to the system tray.

What do I need to do now?

Simply continue to use your computer as normal! We would ask if possible, you inspect the Windows Explorer Quick Access list periodically as the MultiLog software will be managing its contents and we will be asking at the end of the study whether you thought this improved the ways in which you access folders/files on your computer.

Appendix 6: QuickFileAccess Pre-Deployment Survey

The QuickFileAccess Pre-Deployment Survey was designed to gather demographics information on participants and also gain an understanding of how much they believed they utilised the un-modified Windows Explorer Quick Access list.

QuickFileAccess

Q1 Please enter your name:

Q2 Please indicate your gender:

Male (1)

Female (2)

Q3 Please indicate your age:

Under 18 (1)

18 - 24 (2)

25 - 34 (3)

35 - 44 (4)

45 - 54 (5)

55 - 64 (6)

65 - 74 (7)

75 - 84 (8)

85 or older (9)

Q4 Please indicate your employment status:

- Employed (salaried staff) (1)
- Self-employed (2)
- Company director/owner (3)
- PhD student (4)
- Undergraduate student (5)
- Unemployed (6)
- Retired (7)
- Other (please specify) (8)

Q5 If you are related to a University, please state which one and your subject area:

Q6 Please indicate if you use your computer for any of the following (work use):

- E-mail (1)
 - Word processing (2)
 - Spreadsheets (3)
 - Presentation preparation (4)
 - Web browsing (5)
 - Programming (6)
 - Academic paper writing (7)
 - University/school/homework (8)
 - Data analysis (9)
 - Other (please specify) (10)
-
- None (11)

Q7 Please indicate if you use your computer for any of the following (home use):

- E-mail (1)
 - Web browsing (2)
 - Online games (3)
 - Installed (local) games (4)
 - Household admin tasks (such as paying bills) (5)
 - Watching video content (6)
 - Listening to music (7)
 - Voluntary work for organisations (8)
 - Word processing (9)
 - Other (please specify) (10)
-
- None (11)

Q8 If you use your computer for work, please tell us your job title:

The following questions concern the computer the software is/will be installed on.

Q9 What type of computer is the software to run on:

- Desktop (1)
 - Laptop (2)
 - Tablet (3)
-

Q10 Where is this computer usually situated:

- Desktop on desk (1)
 - Laptop on desk (2)
 - Laptop on lap (3)
 - Laptop in mobile location (train, plane etc.) (4)
-

Q11 If you use a laptop, do you use the touchpad or a connected mouse?

- Touchpad (1)
 - Connected mouse (2)
 - Both (3)
 - Not applicable (4)
-

Q12 Does your computer use the same number of screens all the time?

Yes (1)

No (2)

Q13 If yes to question 12, how many screens do you use:

1 (1)

2 (2)

3 (3)

4 (4)

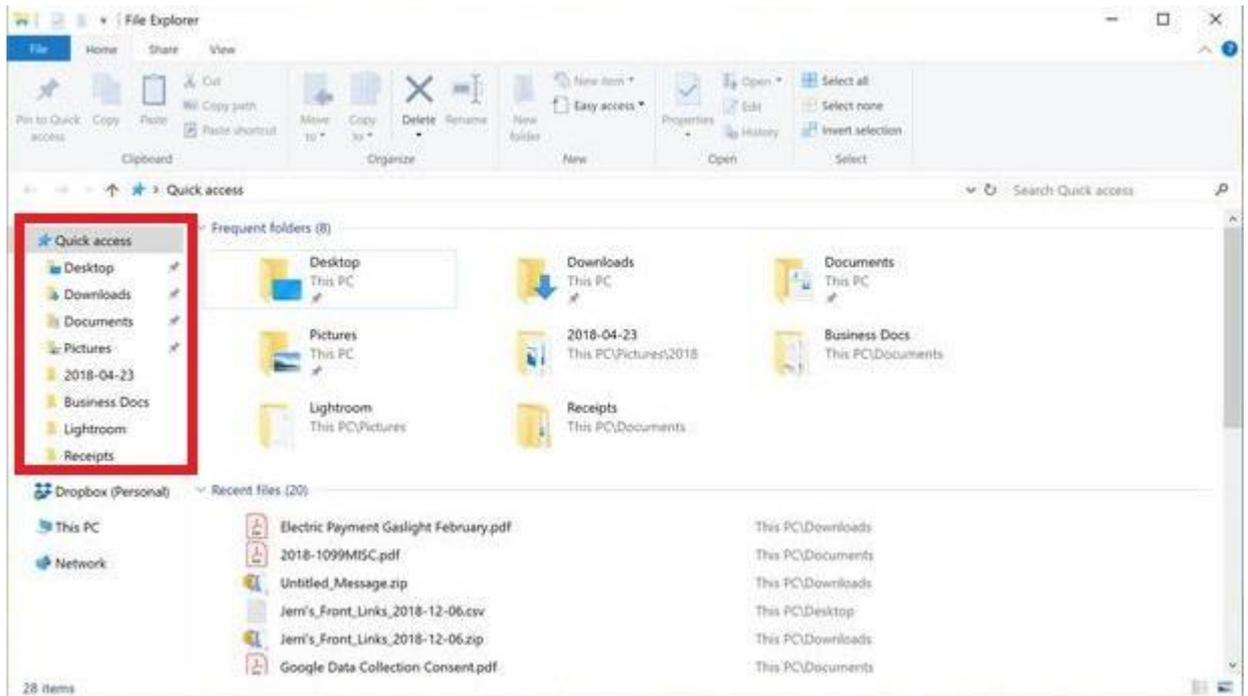
5 or more (5)

Q14

If no to question 16, please describe your screen use (for example "when in the office, my laptop is connected to two additional screens but when mobile or at home, I only use the laptop screen").

The following questions are regarding the Quick Access list in Windows Explorer. Please see the below image for an example screenshot if you are unsure how the Quick Access system works. The Quick Access list is highlighted in red in the image.

Q15

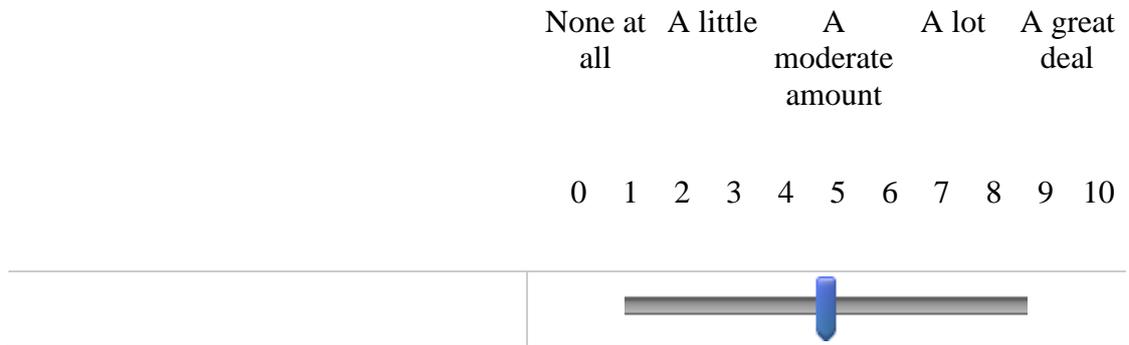


Q16 Do you currently use the Quick Access list in Windows Explorer:

- Yes (1)
 - No (2)
-

Q17

Please indicate how often you use the Quick Access list within Windows Explorer:



Q18 Please indicate the ease with which you can locate files within Windows Explorer:

- Extremely easy (1)
- Moderately easy (2)
- Slightly easy (3)
- Neither easy nor difficult (4)
- Slightly difficult (5)
- Moderately difficult (6)
- Extremely difficult (7)

Q19 Please describe any difficulties you have when locating files on your computer:



Q20 Are there any ways in which you think navigation within Windows Explorer can be improved?

Appendix 7: QuickFileAccess Post-Deployment Survey

The QuickFileAccess Post-Deployment Survey was designed to gather qualitative feedback from users to understand whether the QuickFileAccess tool improved their navigation/decreased the amount of time they spent locating files and folders on their PCs.

MultiLog: Quick Access - End of Study Survey

Q1 This survey should be completed on completion of the MultiLog: Quick Access Study you recently participated in. Please fill in the questions to the best of your ability.

Q2 Please state your name:

Q3 The following questions are about the MultiLog: Quick Access tool. The system manages the Quick Access list in Windows Explorer to present more relevant directories based on your desktop interactions.
The numbers on the scale correspond as follows: 0 = Strongly Disagree 5 = Neither Disagree or Agree 10 = Strongly Agree

0 1 2 3 4 5 6 7 8 9 10

I noticed different directories appeared in the Quick Access list to normal. ()	
The file directories that appeared in the Quick Access list were more relevant to me. ()	
I used the Quick Access list more because file directories which appeared there were more relevant to me. ()	
The tool improved (decreased) the amount of time I generally have to spend looking for files/directories on my computer. ()	

Q4 Please tell us any other feedback you may wish to give regarding the management of the Quick Access list by the MultiLog: Quick Access system.

Appendix 8: MultiLog & QuickFileAccess Software Removal

This document was sent to all participants on completion of the longitudinal log study using the MultiLog software and the pilot study using the QuickFileAccess software to provide step-by-step instructions describing how to remove the MultiLog/QuickFileAccess software from their PCs.

MultiLog – Software Removal Instructions

Thank you!

Firstly, we wish to thank you for your participation in the user study. We have gathered a range of very interesting data which we are in the process of analysing. We may wish to conduct a closing survey. If this is the case, we will contact you again in the coming weeks.

How do I remove the software?

In order to remove the software, please follow the step-by-step instructions below:

Stage 1: Stop the MultiLog Process

1. Check if MultiLog is running in the system tray (this is the notification area located in the bottom right corner of the screen). You will see the “ML” logo if it is running.
2. If it is running, right click the “ML” logo and left click on “Show MultiLog”. The MultiLog interface will be shown.
3. Select “File”, then “Exit” from the menu bar in the MultiLog interface. This will close MultiLog.
4. Check that MultiLog has closed by opening Task Manager. To do this, Press CTRL + ALT + DEL and select “Task Manager”. You can alternatively start it from the Windows Start Menu by searching for “Task Manager”.
5. Inside Task Manager, select the “Processes” tab (click “More Details”, then “Details” tab in Windows 10). Click “Name” to order the process names alphabetically. Check to see if “MultiLog.exe” is running. If it is, right click it and select “End Process”.

Stage 2: Delete Temporary Files

1. Open a Windows Explorer window (search for “Windows Explorer” in the Windows Start Menu if you are not sure how to do this).
2. In the URL/file path bar at the top, navigate to “C:\Users\\Logger\” where “<YOUR USERNAME>” is replaced with your Windows username. You can also click through to “Computer”, then “C:\”, then “Users” in the left hand tree navigation section and the main Explorer window.
3. Delete all files from the “C:\Users\\Logger\” directory. You can also delete the directory/folder itself.

Stage 3: Removing Start Up Commands

1. If you start MultiLog manually every time your computer turns on, ignore this step.
2. If MultiLog starts automatically (you take no action to start it when your computer starts up), follow the instructions below

3. Start the Windows Task Scheduler. Do this by searching for “Task Scheduler” in the Windows Start Menu. It may appear as “Schedule Tasks”.
4. When the Task Scheduler has started, click on “Task Scheduler Library” in the left hand panel.
5. In the top middle panel (which is opened as a result of clicking “Task Scheduler Library”), locate “MultiLog.exe”.
6. When you have located it, right click it and select “Delete”. It should be removed from the list.
7. If “MultiLog.exe” is not contained within the list, you do not need to take any action.

Questions/Further Help

If you require any further assistance with removing the software (or would like a personal visit for me to remove it for you), please do not hesitate to contact me on: jonathan@jonathanwoodruff.co.uk.

Appendix 9: QuickFileAccess Withdrawal

This appendix was sent to participants and explains their rights to withdraw and how to do so should they choose to withdraw.

Withdrawal

Can I withdraw?

Yes, you can withdraw from the study at any time.

How do I withdraw?

Simply notify: jonathan@jonathanwoodruff.co.uk if you want to withdraw.

Will my data be deleted?

Yes, all data will be deleted once your withdrawal request is received.

How do I remove the software?

In order to remove the software, please follow the step-by-step instructions in the debrief sheet.

Questions/Further Help

If you require any further assistance with withdrawal, please do not hesitate to contact: jonathan@jonathanwoodruff.co.uk.