

Making Sense of Low-Level Usage Data to Understand User Activities

KAREN RENAUD AND PHIL GRAY

University of Glasgow

Empirical studies of user activity, based on data collected from the systems with which users interact, present technical challenges related to the transformation of data streams to a form suitable for analysis. In this paper we discuss the particular challenges confronted during a study of user interruption behaviour based on low-level “keystroke” data and the ways in which these challenges were addressed. We also report on a method of data cleaning and analytical preparation that was developed and consider its effectiveness and potential applicability for similar studies.

Categories and Subject Descriptors: H.1.2 [User/Machine Systems]: Human Information Processing; D.2.5 [Testing and Debugging]: Diagnostics, Tracing, Testing Tools, Error Handling and Recovery

General Terms: Low-level data, usage data, transformation, filtering, user activities, flexibility, interruptions, resumption lag activity

1. INTRODUCTION

The GRUMPS project¹ at the University of Glasgow supports exploratory studies of user interaction by collecting low-level usage data and delivering this data to investigators. Investigators typically do not have a specific hypothesis and interrogate and analyse the data in order to uncover specific patterns or interesting sequences of activity. The GRUMPS system supports a feedback loop which allows investigators to request changes in collection activity to support further investigations of such identified patterns of interesting activity.

The principal motivation for the GRUMPS system was to support Rapidly Evolving Data Driven Investigations (RED-Dis) [Thomas et al. 2003; Kennedy and Judd 2003]. Whereas the typical focus of usage data collection is in confirming or denying some pre-conceived hypothesis of application usage, these investigations are characterised by the fact that the question the investigator wishes to answer will change as he/she analyses the data. However, GRUMPS can equally successfully support hypothesis-driven investigations as well as any other usage data collection mechanism, and indeed as a REDDI captures and understands usage data, the investigator will form a hypothesis and use further GRUMPS usage data to verify or deny the hypothesis.

Low-level usage data, such as the data collected by GRUMPS, is a rich source of information for the HCI practitioner. Data is typically captured to reflect the user’s actions involving the keyboard, mouse and other input devices. This data can be augmented by data from cameras or video or even screen dumps which could be triggered by some user action or on-screen activity. Tapping the data provided by these resources, however, is by no means an easy task because of the sheer volume of data and the often intractable nature of the data set [Misanchuk and Schwier 1992; Reeves and Hedberg 2003].

A number of tools exist to support the capture, storage and analysis of this kind of data [Hilbert 2000], but these are mostly aimed at the usability evaluation of one specific application. Traditional exploratory sequential analysis techniques [Fisher and Sanderson 1996] typically start with a research question, and then manipulate observational data in order to answer the questions. Many of these techniques have concentrated on the analysis and modelling of video data [MacKay and Beaudouin-Lafon 1998]. Our investigations, on the other hand, have no fixed question, or thesis, rather collecting a mixture of data describing different events at the user interface to support a variety of diverse investigations. We examine the data in order to find interesting trends or patterns of behaviour. Our approach is to collect as much data as possible. The alternative, and traditional approach, is to collect data about particular events or actions that are considered to be relevant to a particular research question. We, in contrast, collect as much data as possible so that many different investigations can potentially be supported.

This paper will report on an investigation into the user’s overall experience at the user interface, involving all his/her applications, supported by the data collected by the GRUMPS system. We explore the mechanisms for extracting meaningful

¹<http://grumps.dcs.gla.ac.uk>

Author Addresses:

K.V. Renaud, Department of Computing Science, University of Glasgow, 17 Lilybank Gardens, Glasgow, G12 8RZ, United Kingdom; karen@dcs.gla.ac.uk

P.D. Gray, Department of Computing Science, University of Glasgow, 17 Lilybank Gardens, Glasgow, G12 8RZ, United Kingdom; pdg@dcs.gla.ac.uk

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, that the copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than SAICSIT or the ACM must be honoured. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.

© 2004 SAICSIT

information from low-level events recorded by the GRUMPS system.

The investigations can only be carried out if the data can be easily collected and the feedback quickly assimilated into the collection process so as to tailor the resulting data to the particular investigation. This paper describes one such investigation.

2. DATA SET AND INVESTIGATION

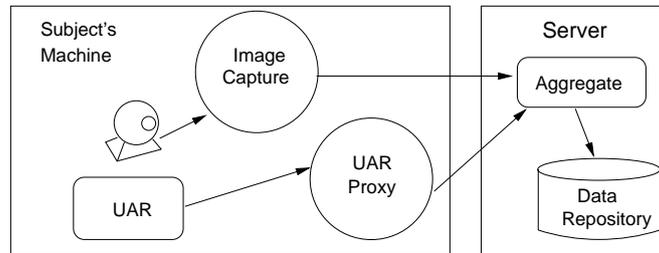


Figure 1. Usage Data Collection

The data being used in the investigation reported in this paper was generated by the User Action Recorder (UAR). This collector uses an elected set of Windows system calls, and gathers large volumes of low-level key events representing actions such as keystrokes and mouse clicks. The UAR was used to collect data from seven subjects over an extended period. The data was stored in a SQL Server database with a simple schema, with entities for only events and sessions. The information is stored in the database in XML format. Three types of events are reported, depending on the way the UAR is configured: window focus events (*wf*), mouse click events (*mc*) and key press events (*kb*).

The GRUMPS usage data is produced by a number of different devices or processes. Raw usage data is thus organised into sequences of usage events. A session is a collection of usage events spanning from the activation of the GRUMPS data collection component to its termination. A session is defined as follows:

```
Raw usage session = <session_id, start_time, end_time, user_id, machine_id, exit_reason>
```

Events belonging to a session are defined as follows:

```
Usage action event: <action_id, session_id, timestamp, type, body>
```

Session and action ids are unique identifiers and the timestamps together with the action identifier provides a natural ordering for events. The body holds one or more of a variety of different details, depending on the collection device and its configuration.

This data is converted, by one of the GRUMPS tools, into *Comma Separated Value* (CSV) data and delivered to the investigator. This file is then converted to a Microsoft Excel file, producing a spreadsheet document such as the one shown in Figure 2.

Once the data is stored in an Excel file, the built-in spreadsheet statistical analysis tools, chart generating tools, and functional capabilities can be deployed. Alternatively tailor-made processes can be developed to perform filtering and transformation operations on the data as shown in Figure 3. Thus the data is filtered, analysed and transformed in a chain-wise fashion, with regular iterations performed as required. After each transformation it is possible for the investigator to inspect the data, to generate a chart or to define a further transformation. Each link in the chain is preserved so that, at any stage, the investigator can backtrack and do a different transformation, pursuing a different theory or hypothesis.

The investigator chose to use Java, and a package called *jexcelapi*² to perform successive transformations on the data. The general idea is illustrated in Figure 3. Each *step* in the chain is a process to refine and filter the *data* in some way. Each *link* in the chain is provided by the spreadsheet files, which encapsulate both the filtered data and the tools to generate the *results*.

All links in the chain are spreadsheets, and can be viewed. This saves the investigator the effort of developing a custom-made viewer for the data. Using spreadsheets has another, equally important, side effect. The processes that are applied to the data in order to: detect events of interest, filter non-events and produce summaries; are as important to the investigation process as the data itself [Foster et al. 2003]. The spreadsheet model enables the investigator to record important provenance information in the same document as the processed data, in a separate sheet, preserving an audit processing trail which is conveniently bound up with the data itself.

²<http://www.andykhan.com/jexcelapi/>

1	A	B	C	E	R	S	T	U	V	W	X	Y	Z	AA
1	type	seq	timestamp	name	p	r	x	y	b	w	wl	wf	wr	wb
14	wf	15	1.08276E+12	2004-3-24T04:40:43	javaw.exe					Event Cact	0	0	533	385
15	mc	16	1.08276E+12	2004-3-24T04:53:190			2379	1008					198	18
16	wf	17	1.08276E+12	2004-3-24T04:55:924	UAR.exe					User Actio	220	220	726	703
17	mc	18	1.08276E+12	2004-3-24T04:56:18			583	343			305	215	370	230
18	cc	19	1.08276E+12	2004-3-24T04:56:18										
19	mc	20	1.08276E+12	2004-3-24T04:41:4:112			2433	1012			0	0	90	18
20	wf	21	1.08276E+12	2004-3-24T04:41:5:987	Explorer.EXE						-2	996	2562	1026
21	mc	22	1.08276E+12	2004-3-24T04:41:8:81			2416	1010			0	1	16	19
22	mc	23	1.08276E+12	2004-3-24T04:41:8:81			2379	1008			0	0	198	18
23	wf	24	1.08276E+12	2004-3-24T04:41:8:UAR.exe						User Actio	-32000	-31840	-31976	
24	wf	25	1.08276E+12	2004-3-24T04:41:8:Explorer.EXE							-2	996	2562	1026
25	wf	26	1.08276E+12	2004-3-24T04:41:1:Explorer.EXE							-2	996	2562	1026
26	mc	27	1.08276E+12	2004-3-24T04:41:16:425			116	925			0	0	2560	996
27	wf	28	1.08276E+12	2004-3-24T04:41:1:Explorer.EXE							-2	996	2562	1026
28	wf	29	1.08276E+12	2004-3-24T04:41:1:Explorer.EXE						Program Iv	0	0	2560	1024
29	wf	30	1.08276E+12	2004-3-24T04:41:1:ftpx.exe						FTP Explo	1102	341	1459	655
30	mc	31	1.08276E+12	2004-3-24T04:41:20:3			1058	629			87	255	162	278
31	wf	32	1.08276E+12	2004-3-24T04:41:2:javaw.exe							-1000	-1000	-1000	-1000
32	wf	33	1.08276E+12	2004-3-24T04:41:2:ftpx.exe						FTP Explo	242	242	2162	971
33	wf	34	1.08276E+12	2004-3-24T04:41:2:ftpx.exe						FTP Explo	885	444	1680	596
34	mc	35	1.08276E+12	2004-3-24T04:41:25:800			908	657			317	93	392	116
35	wf	36	1.08276E+12	2004-3-24T04:41:2:ftpx.exe						FTP Explo	0	242	1920	971
36	wf	37	1.08276E+12	2004-3-24T04:41:2:ftpx.exe						FTP Explo	1124	457	1444	583
37	wf	38	1.08276E+12	2004-3-24T04:41:2:ftpx.exe						Options	816	559	1165	946
38	hkb	39	1.08276E+12	2004-3-24T04:41:27:635										
39	hkb	40	1.08276E+12	2004-3-24T04:41:27:722										
40	hkb	41	1.08276E+12	2004-3-24T04:41:27:910										
41	hkb	42	1.08276E+12	2004-3-24T04:41:28:285										
42	hkb	43	1.08276E+12	2004-3-24T04:41:28:441										
43	hkb	44	1.08276E+12	2004-3-24T04:41:28:785										
44	hkb	45	1.08276E+12	2004-3-24T04:41:29:19										
45	hkb	46	1.08276E+12	2004-3-24T04:41:29:207										

Figure 2. Spreadsheet containing Usage Data

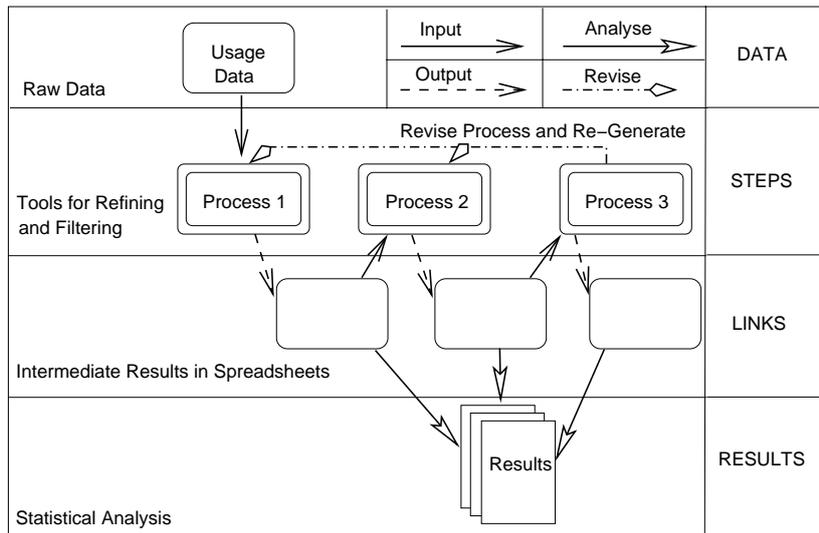


Figure 3. General Chain Structure of Transformation

3. INTERPRETATIONAL CHALLENGES

The investigation was interested in interruptions — cessations of activity — and specifically in user activity *after* the interruptions. The investigator wished to gain an understanding of the actions the user took to re-establish mental context after an interruption. To this end, it was necessary to find such cessations in activity, and to examine user activity straight afterwards to determine how users recover from interruptions. The following subsections will discuss challenges faced in trying to achieve this. These challenges are related to the nature of the low-level usage data in general and to data collected by Windows system calls in particular.

3.1 Identifying Events of Interest and Dealing with Phantoms

At first glance this is a very simple transformation: one simply compares the time stamp of each activity to the time stamp of the preceding activity, and highlight those events which occur after a time lapse. When this process was executed, and the interruption intervals identified, the investigator noticed a prevalence of 10-minute interruptions. A further transformation was carried out in order to find out whether the activity being engaged in immediately before an interruption, was resumed after the interruption. The output from this transformation showed that most 10 minute interruptions were followed by apparent activity in an application with a name such as `logon.scr`, which is actually a screensaver. Thus the apparent resumption of activity was a phantom activity, initiated by a process that kicked in after a time lapse. When the user *did* resume activity after the screen saver had engaged, this was signalled by the appearance of `taskmgr.exe`. Thus the first

step had to be re-factored in order to ignore these phantom resumptions of activity and to detect only genuine resumptions.

When the second step generated revised output, the resumption in activity appeared more meaningful. This experience made it clear that one cannot assume that the user has initiated activity being reported by the UAR. Some window focus events are generated by the underlying system, and any effort to understand user activity needs to screen out these phantom events.

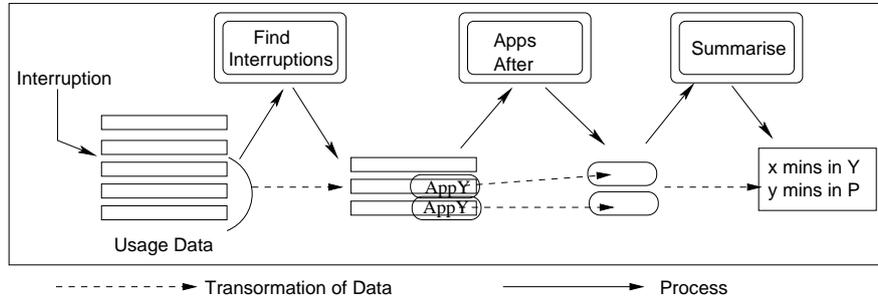


Figure 4. Initial Filtering Process

3.2 Establishing Context

The third step was now developed to calculate the number of actions the user carried out within each application, and to summarise the actions taken in the first few minutes after resumption of activity. This, once again, was much harder to calculate than it first appeared to be. The main problem was one of establishing context. Low-level usage events, in isolation, typically do not carry enough information to unambiguously establish the application to which it should be related. Since events are essentially sequential in nature, the most obvious approach appeared to be that a transformation would be applied with the following rule:

```

while (not end of file) {
  read event
  if event is window focus, remember window title and process name
  if event is mouse click or keypress, attribute to process
  evidenced by most recent window focus event
}
    
```

Unfortunately this approach was flawed. For example, an event, such as a mouse click, was initially attributed to the most recent window focus process. Consider that a user may have two applications active: Outlook to read email, and Internet Explorer to browse the web. The user may currently be in Outlook and decide to switch to Internet Explorer. She may do this by clicking on the title bar. The UAR will record a mouse click, and then a window focus event for `iexplore.exe`. A simplistic propagation of context from the previous window focus event would classify the mouse click as belonging to Outlook whereas in reality it should be assigned to Internet Explorer.

The only way to deal with this is to examine the time stamps of subsequent events so that events with the same time stamp, such as the mouse click and window focus referred to in the previous example, are related to one another.

Another problem is related to startups of new applications. A user may start an application from a desktop icon, or from the Start menu. Invoking the start menu and choosing Internet Explorer generates 6 different events, and the pattern is not always the same. One pattern is shown in Table I. However, the first two events are sometimes reported in reverse order because the event time stamp is added subsequent to the event and may be generated in a different order from actual event ordering.

Action Type	Window Title	Process
Mouse Click		
Window Focus		explorer.exe
Window Focus	start menu	explorer.exe
Mouse Click		
Window Focus		
Window Focus		explorer.exe

Table I. Pattern of Activity for Starting An Application from the Start Menu

This sequence of events is translated and replaced with a single event, of type “Activate Explorer from Start Menu”, adding a level of interpretational data to support understanding of user activity. The other problem is related to users switching between applications using the Alt-Tab key combination. This key combination should not be attributed to any active application and is important in terms of this investigation, since it should possibly signal an internally-generated interruption and switching of attention to another application. This could be done in the context of the user’s overall task or it could be totally unrelated to what the user is currently doing.

This activity is distinctly different from the one described previously where mouse clicks need to be assigned to the correct application. The cognitive context is different here and must be interpreted differently — ie. not attributed to any particular application.

There is no way to infer this from raw usage data. In searching for these actions we discovered differences between users. Some users exhibited Alt-Tab actions whilst others exhibited only successions of Alt keys. After observing the users it was found that the former users would press the Alt key, then press the Tab key and then release both. The latter users would hold the Alt key down and use the Tab key to step through active applications in order to choose the ones they wanted. The different usages are shown in Figure 5. The activity within this menu was not recorded by the UAR. Thus,

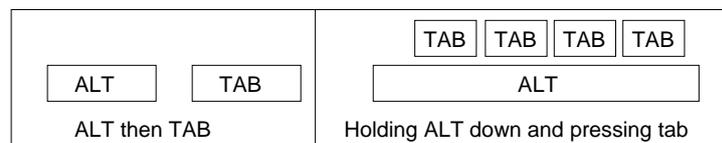


Figure 5. *Alt-Tab Combinations*

once again, we had an ambiguous situation: whereas Alt-Tab was easy to detect, the second type of usage was difficult to pin down since the Alt could be related to the following key press, such as the use of Alt-S in Emacs, and it could play different roles in other applications. The investigator is faced with a dilemma: the best way to handle this kind of investigation is to treat applications generically and not to use application semantics to interpret usage data. To detect this kind of Alt-generated switching we need to sacrifice genericity.

A related problem occurs when one application launches another. Outlook regularly launches `winword.exe` to deal with emails. The window focus events, at first glance, appear to indicate that a new application has been launched, whereas the activity that takes place within this launched application should actually be attributed to Outlook and not Word. It is necessary to be specific when interpreting this kind of event. This particular problem can only be resolved by examining the window titles and drawing conclusions about the nature of the activity being engaged in.

If the window title is examined we can refine our understanding of activities within Outlook. One can detect whether an email is being created, replied to, printed or forwarded. This is important if one wants to investigate time spent dealing with emails during a session.

The next step in the investigative chain, once the context had been propagated, was to determine how many activities took place within each application.

3.3 Dealing with Phantom or Missing Actions

In order to understand user activity and interaction with the user interface the next step in the chain was to count the number of actions in each application. When this step was executed the results were plotted on a graph. It immediately became clear that the number of keystrokes being reported was unrealistically large.

We found that some keys would, when held down, generate a flurry of UAR events. This would happen particularly with the Alt and Shift keys, which are often held down for extended periods. In order to understand the user activity these repetitive and superfluous actions need to be filtered out so that they do not skew the results of the investigation. Thus a new step was inserted right at the beginning of the analysis. This step “cleans” the data, removing duplicate key presses. However, when doing this one has to ensure that intentional key presses are retained. Thus repeated backspaces are probably intentional, as are repeated alphanumeric key presses.

Another problem, which is far more difficult to deal with, is that of “missing” events. For example, when the user closes an application window by using the Alt-F4 key, this is detected and reported by the UAR. If the user clicks on the X at the upper right-hand corner of the window, no event is reported. Thus it becomes impossible to maintain a tally of active applications: firstly because the UAR starts recording at arbitrary times and other applications are probably already active, and secondly because we cannot know when application windows are closed. Thirdly, it is difficult to distinguish between duplicate windows of the same application, as is common with Internet Explorer. When the window is first generated a window focus event is recorded, and the window location can be detected from this event. However, when the user moves one of these windows to a new location no event is reported so the previously detected location becomes useless in determining what is happening on the screen. The only way to determine this is to trigger screen shots either at

regular intervals or when the user carries out some defined pattern of actions. The process chain generated to deal with the anomalies discussed in this section is shown in Figure 6.

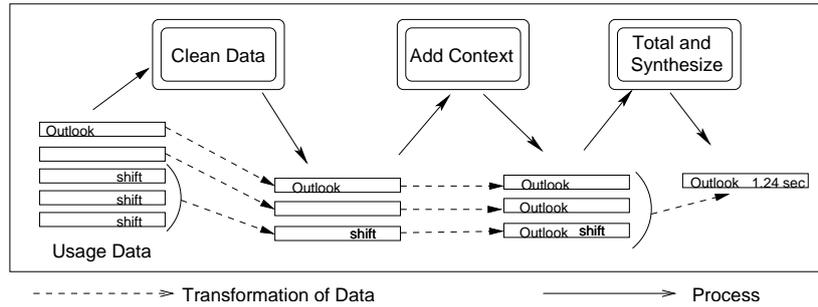


Figure 6. Usage Data Collection

4. INTERRUPTION STUDY

Interruptions are a fact of life. Many people feel that they are detrimental [Bailey et al. 2001] but studies have shown that some interruptions do have positive effects [van Solingen et al. 1998] and indeed when the user is involved in a monotonous activity an interruption can have a positive effect on performance [Cabon et al. 1990]. Jett and George [Jett and George 2003] classify interruptions into four types: intrusions, breaks, distractions and discrepancies. Whatever the cause of the interruption, the user is faced with having to resume the primary task after the interruption has been processed. The structure of an interruption is depicted in Figure 7. The length of the resumption lag depends on many factors, including the type of interruption, the age of the user, the number of active tasks, how knowledge-intensive the tasks are, the frequency of interruptions and the support provided by the user interface [Monk et al. 2002; McFarlane 1997; Miller 2002; Franke et al. 2002; Card and Henderson 1987].

Previous studies have found that the activity engaged in during the interruption lag will affect the length of the resumption lag [Altmann and Trafton 2004; Miller 2002]. There are also important differences between internal (self-generated) and external interruptions. So, for example, a user may be busy working on a Word document and then decide to check email: an internal interruption. An external interruption may come from a telephone call, a knock on the door or a program that signals the arrival of an email message. The difference lies in the users' ability to anticipate and prepare for the interruption of their primary task during the interruption lag.

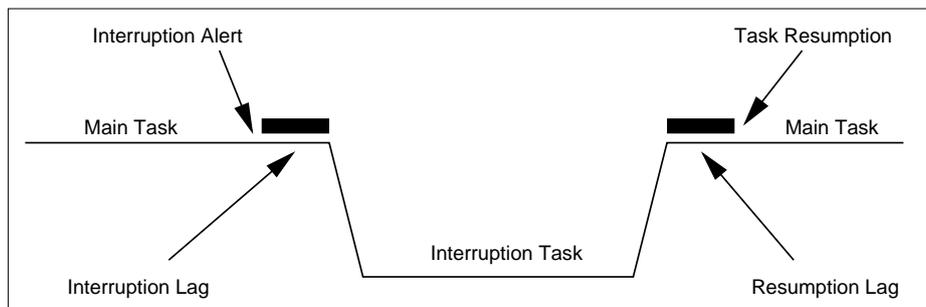


Figure 7. Structure of an Interruption

The interruption study set out to find out how computer users behaved during the resumption lag as contrasted to their general behaviour during an entire session. This investigation works with raw usage data, and thus our understanding of the user's actions and tasks is somewhat curtailed. In order to examine resumption lag activity the procedure demonstrated in Figure 4 was used to extract usage data relating to activity relating to the first 10 minutes after each interruption. Thereafter the process demonstrated in Figure 6 was applied to the original file, and to a summary of the post-interruption behaviour activity. The following data was produced:

- The time spent doing different activities in each application. This is quite coarse-grained in some applications such as Internet Explorer where no attempt has been made to distinguish the semantics of different activities within the application. In Outlook, however, a more fine-grained analysis has been carried out since it is possible to derive a great deal from examination of Window titles.
- The number of user actions (key presses and mouse clicks) attributed to each application activity.

- The minimum time spent doing each application activity.
- The maximum time spent doing each application activity.
- The average time spent doing an activity.
- How often the user engaged in an activity during the entire session, derived by dividing the total time by the number of times the application activity was engaged in.
- Percentage of overall time spent doing each activity.
- Percentage of actions performed doing each activity.
- The frequency of switching activity — the activity switched from and the activity switched to.
- The application used during the interruption lag and during the resumption lag.

Some interesting results are emerging for the subjects currently being monitored. For example, we were able to deduce that:

- (1) users resumed the same application after an interruption 55% of the time (on average), with a standard deviation of 19.23%.
- (2) users checked their email messages straight after an interruption 52% of the time and checked their email *before* an interruption 47% of the time — suggesting that the interruption lag was being used productively in some cases.
- (3) users tended to make slightly fewer typing errors after an interruption and typed slightly faster too (average 10ms faster). The latter finding may suggest a sense of urgency after an interruption.
- (4) users who are involved in teaching activities are more likely to check email straight after an interruption. Many interpretations could apply to this — it could be a consequence of increasing student reliance on email as a communication mechanism and lecturer response to this; it could be an indication of the pressure email is considered to apply; or it could be merely a personal habit of the academics involved in this study. Without a self-assessment questionnaire it is impossible to answer this question.
- (5) there are large variations over time both between different users and between users in different recorded sessions. This is related to the particularly diverse nature of the activities engaged in by users. However, if the activity sequences for user sessions are compared using a pattern-matching algorithm [Irving 2004] a greater similarity between a particular user's different sessions is observed than between that user and other users. The similarity count, determined using the Smith-Waterman algorithm [Smith and Waterman 1981], is given in Table II. The similarity difference is not large enough to be significant but it appears that different users have different and distinct application usage patterns. This may be of interest to the recommender systems community, who could perhaps make use of this kind of information to make recommendations to other users. Furthermore, this information could of use to the operating system in deciding which applications to hibernate or activate based on the likelihood of subsequent usage. For example, an analysis of one particular user's session is shown in Figure 8. The numbers on the axes represent applications, with the mapping given in the table on the right hand side. The colour-graduated bar to the right of the graph gives a colour mapping which represents the probability of a switch to a particular application (x axis) from the application currently being used (y axis). So, for example, for this particular user we see that activity in Windows explorer is likely to be followed by an email check with a probability of 37% whereas the likelihood of a switch to an email check from Microsoft Word is very low.

	User 1	User 2	User 3	User 4	User 5
User 1	44	16	9	37	4
User 2	16	44	37	14	3
User 3	9	37	45	17	1
User 4	37	14	17	46	10
User 5	4	3	1	10	26

Table II. Similarity Between Users' Sessions

This investigation confirmed the wisdom of using spreadsheets to store intermediate data. When the investigator needed additional data to clarify differences it was trivial to add a new column to derive the new data without having to code the process. However, the investigation also emphasised the difficulties related to investigations of this type, where an attempt is made to gain an understanding of high-level activities from low-level data. For example, it is impossible to tell whether a cessation in activity is caused by an interruption or perhaps by the fact that the user has completed her task. It is also impossible to determine whether an interruption is internal or external or which of Jett and George's categories it belongs to. It is also impossible to derive any measure of an overarching user task which involves the use of various applications.

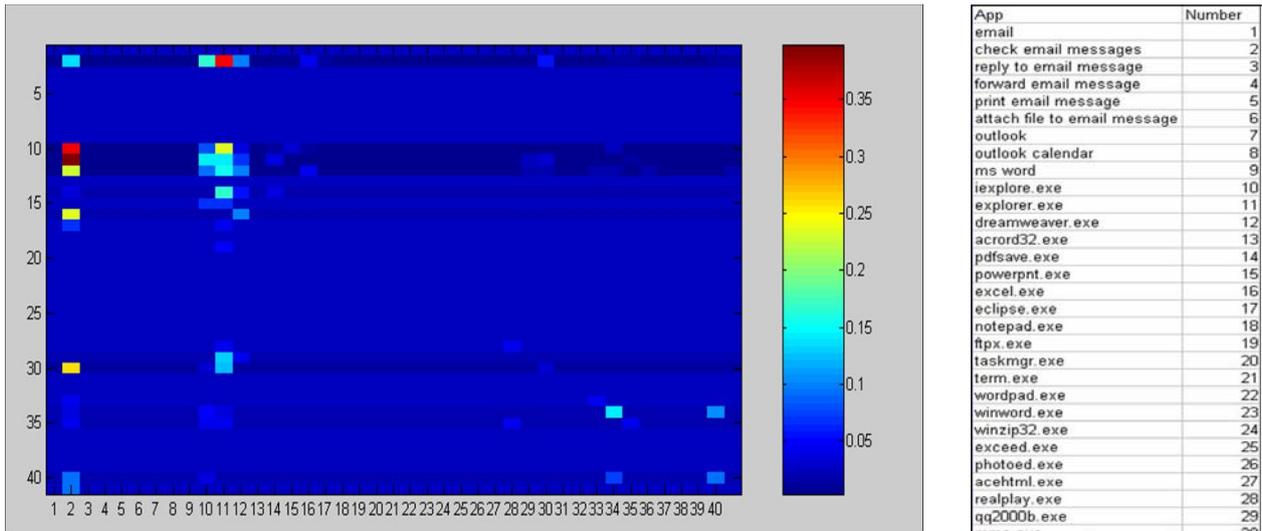


Figure 8. Probability of Next Application Use

For example, a switch from email to explorer could denote an interruption but could also be part of a single task where the user is saving an email attachment.

Furthermore, it is impossible to gather exactly what the user is doing during the resumption lag using only low-level usage data. Users re-establish context by using cues offered by the user interface, but we cannot know which cues are being used. They may also have made written notes to help them to resume more easily. Low-level usage data cannot help an investigator to gain insight into this activity. However, the low-level usage tool does give the user an insight into the kinds of activities being engaged in, and how the user makes use of various applications, and acts as a spring-board to launch further investigation.

In conclusion, the use of a low-level tracking tool such as GRUMPS is ideal as an *initial* tool to help an investigator to formulate research questions — but it cannot be used to infer much about high-level cognitive activity. The investigation should then progress to the use of a higher-level tool, as described in Section 5, in order to refine the investigation and to gain a better understanding of user behaviour.

5. OTHER POSSIBLE INVESTIGATION MECHANISMS

The issues confronted in this study were particularly acute, given the rather large gulf between the nature of the data collected and the level of abstraction required to answer the investigation research questions. One obvious way of alleviating the difficulties would be to reduce this semantic gulf. There are several ways in which this could be done, and various alternatives are discussed in the next section.

5.1 Instrumenting Higher-level Triggers

This approach identifies triggers that are closely correlated with the desired behaviours of a particular application and collect these trigger events directly. For example, one could construct a usage monitor that generates an event when a gap of processing of more than 10 seconds occurs, which could indicate that the user is having a problem determining what actions are possible at that particular stage in the dialogue.

This has been done successfully by various people, for example the work done by [Welland et al. 1997], but it may require access to the source code of the application. It also assumes that the trigger events are known and that we can identify them correctly and capture them reliably. One can expect that this would be a follow-on stage from the earlier exploratory data collection when an understanding of the the associations between triggers and behaviour emerges from analysis of the raw data.

5.2 Capturing more context

One could collect additional contextual data in order to remove phantom actions and identify higher level actions. An example of this is the use of screen scraping and video footage of the user as he or she works. However, there is a cost-benefit trade-off which needs to be considered. The data collection costs and the associated effort required to make the connection between usage data and context data needs to be weighed against the potential benefits of increased context data and improved understanding and interpretation of user activity. The cost in this case is fairly extensive since screen shots are large and the regular transmission of these would have to be very valuable before this clogging up of the bandwidth and slowing down of the user’s machine can be justified.

5.3 Involve the User

In the case of human-based usage monitoring it is possible to involve the user in the interpretive process. This might be possible in an early exploratory or piloting stage even if it were important to avoid such disruptions in subsequent data collection. However, the Hawthorne effect [Roethlisberger and Dickson 1939] may skew the results if the user is explicitly involved. The validity of Hawthorne's findings have been questioned by researchers using statistical techniques [Franke and Kaul 1978]. These researchers argue that the altered behaviour may not have been caused solely by the observation process, as Hawthorne believed, but rather by the fact that productive and disciplined managerial workers replaced insubordinate and mediocre workers. Franke and Kaul believe that this was the causative in this effect and not merely the observation process.

Even without this complication there is another problem related to involving the user: the time consuming nature of the investigation. It is likely that user involvement will limit the study to users who are interested in this kind of research and will not be able to investigate a variety of different users.

6. ISSUES AND CAVEATS

6.1 Ethics and Obfuscation

In doing investigations of this type there is a continuous balance between ethics and the needs of data analysis and mining. One has to find the right balance but it is not always obvious where the balance is at the outset of the investigation.

One can record all activity and this potentially constitutes a major security risk. Users will often enter passwords and use their browsers for online banking, and the UAR should not record such information. However, some information is extremely important in order to make sense of activity at the user interface: to compose higher order actions from recorded usage data. Our initial study obfuscated all key presses so that user privacy was completely protected. This was done at the client machine, so that such an event would simply be recorded as "hidden key press".

However, it became clear as the analysis of the data proceeded that some data would need to be revealed to support analysis of the data. GRUMPS was designed with this kind of feedback loop in mind [Evans et al. 2003], and thus the UAR was adjusted so that only alphanumeric, numeric and special character data was obfuscated — as depicted in Figure 9. All modifier data, such as Alt, Ctrl, Shift and Tab, was reported without obfuscation so that more detailed and meaningful analysis could be facilitated.

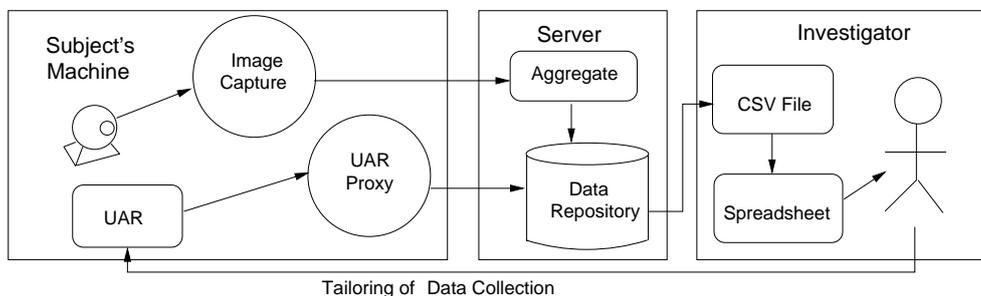


Figure 9. *Tailoring the Collection*

6.2 Resource Constraints

In a study such as this one, when one has decided to make use of the low-level data collection approach, it is necessary to collect as much data as possible. The more data we have, the more understanding we can gain of user activity. However, there is a balance between data collection and resource constraints. One cannot collect every single mouse movement, for example, since a simple navigation from the top of the screen to the bottom would potentially generate hundreds of mouse movement events. The investigator could easily "drown" in this data and have to run a cleaning process to remove it.

Sending too much data may also impact negatively both on the user's machine on the network, and on the GRUMPS collection system. Thus we have to find the happy medium, between having enough data to investigate a particular type of user interface issue, and minimising the impact on the network and the user.

7. CONCLUSION

Processing low-level usage data is by no means a solved problem. Modern software technology, such as, for example, the JMX package in the latest release of Java³, has recently provided an enabling technology for collecting low-level data

³<http://java.sun.com>

similar to the data discussed in this paper. The emergence of this technology will, we believe, precipitate an exponential increase in monitoring activity and the associated generation of masses of low-level usage data.

This makes gaining an understanding of the problems related to data cleaning and interpretation even more important. Interpreting this data appears, at first glance, to be a simple task, but our experience suggests the contrary. The data needs to be filtered and cleaned so that we derive data which can be dealt with using traditional data analysis techniques such as statistical analysis and data mining. The key to this processing lies in the provision of tools which support the cost-effective cleaning and understanding of the data. The tools and methodology reported in this paper provide a first step towards satisfying this need.

ACKNOWLEDGMENTS

We acknowledge the contributions of Iain McLeod and Iain Darroch, without whom this work could not have been carried out.

REFERENCES

- ALTMANN, E. M. AND TRAFTON, J. G. 2004. Task interruption: Resumption lag and the role of cues. In *Proceedings of the 26th annual conference of the Cognitive Science Society*. Erlbaum, Hillsdale, 42–47.
- BAILEY, B. P., KONSTAN, J. A., AND CARLIS, J. V. 2001. The effects of interruptions on task performance, annoyance, and anxiety in the user interface. In *Human-Computer Interaction - INTERACT 2001 Conference Proceedings*, M. Hirose, Ed. IOS Press, IFIP, Tokyo, Japan, 593–601.
- CABON, P., COBLENTZ, A., AND MOLLARD, R. 1990. Interruption of a monotonous activity with complex tasks: Effects of individual differences. In *Proceedings of the Human Factors Society 34th Annual Meeting*. Orlando, Florida, 912–16.
- CARD, S. K. AND HENDERSON, A. 1987. A multiple, virtual-workspace interface to support user task switching. *ACM SIGCHI Bulletin* 17, 53–59.
- EVANS, H., ATKINSON, M., BROWN, M., CARGILL, M., CREASE, M., DRAPER, S., GRAY, P. D., AND THOMAS, R. C. 2003. The pervasiveness of evolution in GRUMPS software. *Software: Practice and Experience* 33, 2 (feb), 99–120.
- FISHER, C. AND SANDERSON, P. 1996. Exploratory sequential data analysis: exploring continuous observational data. *Interactions* 3, 2 (March), 25–34.
- FOSTER, I., VOCKLER, J., WILDE, M., AND ZHAO, Y. 2003. The Virtual Data Grid: A New Model and Architecture for Data-Intensive Collaboration. In *Proceedings CIDR Conference*. Asilomar CA.
- FRANKE, J., DANIELS, J., AND MCFARLANE, D. 2002. Recovering context after interruption. In *24th Annual Meeting of the Cognitive Science Society*. CogSci. Fairfax, VA.
- FRANKE, R. H. AND KAUL, J. D. 1978. The Hawthorne experiments: First statistical interpretation. *American Sociological Review* 43, 623–643. <http://www.nwlink.com/~donclark/hrd/history/hawthorne.html>.
- HILBERT, D. M. 2000. A survey of computer-aided techniques for extracting usability information from user interface events. *ACM Computing Surveys* 32, 4, 384–421.
- IRVING, R. 2004. Plagiarism and collusion detection using the Smith-Waterman algorithm. Tech. Rep. TR-2004-164, University of Glasgow, Computing Science Department Research Report.
- JETT, Q. R. AND GEORGE, J. M. 2003. Work interrupted: A closer look at the role of interruptions in organizational life. *Academy of Management Review* 28, 3, 494–509.
- KENNEDY, G. AND JUDD, T. 2003. Iterative analysis and interpretation of audit trail data. In *Proceedings of the 20th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE)*. Adelaide, Australia.
- MACKEY, W. E. AND BEAUDOUIN-LAFON, M. 1998. DIVA: Exploratory Data Analysis with Multimedia Streams. In *Proceedings CHI 1998*. ACM Press, Los Angeles, CA, 416–423.
- MCFARLANE, D. C. 1997. Interruption of People in Human-Computer Interaction: A General Unifying Definition of Human Interruption and Taxonomy. Tech. Rep. NRL/FR/5510-97-9870, NRL Formal Report, Washington: US Naval Research Laboratory.
- MILLER, S. L. 2002. Window of opportunity: Using the interruption lag to manage disruption in complex tasks. In *Proceedings of the 46th Annual Meeting of the Human Factors and Ergonomics Society*. Human Factors and Ergonomics Society, Santa Monica.
- MISANCHUK, E. R. AND SCHWIER, R. 1992. Representing interactive multimedia and hypermedia audit trails. *Journal of Educational Multimedia and Hypermedia* 1, 3, 355–372.
- MONK, C. A., BOEHM-DAVIS, D. A., AND TRAFTON, J. G. 2002. The attentional costs of interrupting task performance at various stages. In *Proceedings of 46th Annual Meeting of the Human Factors and Ergonomics Society (HFES 2002)*. Human Factors and Ergonomics Society, Santa Monica, 1824–1828.
- REEVES, T. C. AND HEDBERG, J. G. 2003. *Interactive Learning Systems Evaluation*. Educational Technology Publications, Englewood Cliffs, NJ.
- ROETHLISBERGER, F. J. AND DICKSON, W. J. 1939. *Management and the Worker*. Harvard University Press, Cambridge, Mass.
- SMITH, T. F. AND WATERMAN, M. S. 1981. Identification of common molecular sub-sequences. *Journal of Molecular Biology* 147, 195–197.
- THOMAS, R., KENNEDY, G., DRAPER, S., MANCY, R., CREASE, M., EVANS, H., AND GRAY, P. 2003. Generic usage monitoring of programming students. In *Proceedings of the 20th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE)*. Adelaide, Australia.
- VAN SOLINGEN, R., BERGHOUT, E., AND VAN LATUM, F. 1998. Interrupts: Just a Minute Never Is. *IEEE Software* 15, 5, 97–103.
- WELLAND, R., SJOBERG, D., AND ATKINSON, M. 1997. Empirical analysis on automatic tool logging. In *Empirical Assessment in Software Engineering*. University of Keele.