Supporting Focused Work on Window-Based Desktops

by

Jan Pilzer

B.Sc., DHBW Mannheim, 2014

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL
STUDIES

(Computer Science)

The University of British Columbia

(Vancouver)

October 2019

© Jan Pilzer, 2019
The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the thesis entitled:

Supporting Focused Work on Window-Based Desktops

submitted by Jan Pilzer in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Examining Committee:

Reid Holmes, Computer Science  
Co-Supervisor

Thomas Fritz, Computer Science  
Co-Supervisor

Gail Murphy, Computer Science  
Supervisory Committee Member
Abstract

When working with a computer, information workers continuously switch tasks and applications to complete their work. Given the high fragmentation and complexity of their work, staying focused on the relevant pieces of information can become quite challenging in today’s window-based environments, especially with the ever increasing size of display technology. To support workers in staying focused, we conducted a formative study with 18 professional information workers in which we examined their computer based and eye gaze interaction with the window environment and devised a relevance model of open windows. Based on the results, we developed a prototype to dim irrelevant windows and reduce distractions, and evaluated it in a user study. Our results show that participants keep an average of 12 windows open at all times, switch windows every 17 seconds, and that our prototype was able to predict and highlight relevant open windows with high accuracy and was considered helpful by the users.
Lay Summary

When working with a computer, information workers continuously switch tasks and applications to complete their work. Given the high fragmentation and complexity of their work, staying focused on the relevant pieces of information can become quite challenging in today’s computer environments, especially with the ever increasing size of screens. To support workers in staying focused, we conducted a formative study with 18 professional software developers in which we examined their computer and eye gaze interaction with their computer windows and devised a relevance model of open windows. Based on the results, we developed a prototype to dim irrelevant windows and reduce distractions, and evaluated it in a user study. Our results show that participants keep an average of 12 windows open at all times, switch windows every 17 seconds, and that our prototype was able to determine and highlight relevant open windows with high accuracy and was considered helpful by the users.
Preface

All of the work presented henceforth was conducted in the Software Practices Laboratory at the University of British Columbia and the Software Evolution & Architecture Lab of the University of Zurich. All projects and associated methods were approved by the University of British Columbia’s Research Ethics Board [certificates #H17-02682 and #H19-01449].

A version of this material has been submitted to be published [Jan Pilzer, Raphael Rosenast, André N. Meyer, Thomas Fritz, and Elaine M. Huang. Supporting Focused Work on Window-Based Desktops].

The study and some initial analysis in Chapter 3 was conducted by Raphael Rosenast in Zurich and included in his Master’s thesis. I received his anonymized data and performed additional detailed analysis tailored to my research goals. I was the lead investigator for the projects located in Chapter 4 and 5, responsible for all major areas of concept formation, data collection and analysis, as well as manuscript composition and participant recruitment. Thomas Fritz was the supervisory author on this project and was involved throughout the project in concept formation and manuscript composition. André N. Meyer and Elaine M. Huang contributed to manuscript edits.
# Table of Contents

Abstract ................................................................. iii

Lay Summary ........................................................... iv

Preface ................................................................. v

Table of Contents ...................................................... vi

List of Figures ........................................................ ix

1 Introduction ......................................................... 1

2 Related Work ....................................................... 4
   2.1 Window Interaction Studies ................................. 4
   2.2 Detecting Relevance ........................................ 6
   2.3 Supporting Focused Work ................................... 6

3 Study 1: Developers’ Window Interactions ........................ 8
   3.1 Monitoring Study ............................................ 8
      3.1.1 Participants ............................................. 8
      3.1.2 Study Method .......................................... 9
      3.1.3 Monitoring Application ............................... 9
      3.1.4 Collected Data ....................................... 10
   3.2 Results ....................................................... 11
      3.2.1 Open Windows Behavior ............................. 11
      3.2.2 Window Switching Behavior ......................... 12
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.3</td>
<td>Multi-Monitor Usage</td>
<td>13</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Usage of Screen Real Estate</td>
<td>14</td>
</tr>
<tr>
<td>3.2.5</td>
<td>Visual Attention and Focus on Windows</td>
<td>15</td>
</tr>
<tr>
<td>3.2.6</td>
<td>Behavior by Activity</td>
<td>15</td>
</tr>
<tr>
<td>3.3</td>
<td>Discussion</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>Predicting Relevant Windows</td>
<td>18</td>
</tr>
<tr>
<td>4.1</td>
<td>Model</td>
<td>18</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Temporal, Recency</td>
<td>18</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Temporal, Duration</td>
<td>19</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Temporal, Frequency</td>
<td>19</td>
</tr>
<tr>
<td>4.1.4</td>
<td>Semantic, Window Title</td>
<td>19</td>
</tr>
<tr>
<td>4.2</td>
<td>Empirical Analysis based on Study 1</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>Study 2: Highlighting Relevant Windows to Support Focus</td>
<td>22</td>
</tr>
<tr>
<td>5.1</td>
<td>WindowDimmer Approach</td>
<td>22</td>
</tr>
<tr>
<td>5.2</td>
<td>Intervention Study</td>
<td>24</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Procedure</td>
<td>25</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Participants</td>
<td>26</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Collected Data</td>
<td>26</td>
</tr>
<tr>
<td>5.3</td>
<td>Results</td>
<td>26</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Window Interaction Behavior</td>
<td>27</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Predicting Relevant Windows</td>
<td>27</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Evaluation of WindowDimmer</td>
<td>28</td>
</tr>
<tr>
<td>5.4</td>
<td>Participant Feedback</td>
<td>30</td>
</tr>
<tr>
<td>6</td>
<td>Threats to Validity</td>
<td>32</td>
</tr>
<tr>
<td>6.1</td>
<td>Construct Validity</td>
<td>32</td>
</tr>
<tr>
<td>6.2</td>
<td>Internal Validity</td>
<td>33</td>
</tr>
<tr>
<td>6.3</td>
<td>External Validity</td>
<td>33</td>
</tr>
<tr>
<td>7</td>
<td>Discussion</td>
<td>35</td>
</tr>
<tr>
<td>8</td>
<td>Conclusion</td>
<td>38</td>
</tr>
</tbody>
</table>
List of Figures

Figure 3.1 Average number of open and visible windows by participant. 12
Figure 3.2 Weighted mean of open windows by participant and hour during a typical work day where we could gather enough data from all participants. 13
Figure 3.3 Time between window switches. The bump at 300 seconds is caused by our 5 minute idle timeout. The last bar at 600 represents all longer times between window switches. 14
Figure 4.1 Probability of predicting the correct next window within the top X most relevant windows. 20
Figure 5.1 WindowDimmer dims all less relevant windows and the desktop to emphasize the top 3 most relevant windows. 23
Figure 5.2 WindowDimmer settings that allow enabling, switching the mode for the study, and setting the number of windows that don’t get dimmed. 24
Figure 5.3 WindowDimmer pop-up asking participants to select the relevant windows out of a list of all open windows. 25
Figure 5.4 Number of windows reported (by participants) and predicted (by our model) relevant averaged over all reports. The three top scoring windows in our model are considered relevant. 28
Figure 5.5  Distribution of the duration a window is active before and after the dimming is activated by type of application. The distribution is binned into 3 buckets (0-1s, 1-5s, 5+s) to highlight the changes. 29

Figure 5.6  Percentage of desktop area dimmed by desktop size. Both are calculated across all screens. 30
Chapter 1

Introduction

Multi-tasking and fragmentation of work are known challenges of modern knowledge work [9, 15, 32]. While multi-tasking is necessary and beneficial in enabling information workers to make progress on more than one task, it can come at a significant cost: reduced quality and more errors [43], more time overall [42, 43], increased stress [2, 26], and lower productivity [28, 31]. Although these difficulties are to some extent inherent and inevitable side effects of engaging in multiple work activities simultaneously, we believe that these effects can also be decreased if the information worker can focus only on the most relevant tasks at any given time, and is subjected to limited distraction from less relevant tasks.

Modern window-based computer desktops provide support for multi-tasking by allowing information workers access and visibility to many applications and digital artifacts simultaneously. However, with increasing screen size, multi-monitor support and the ability to run several applications and open many windows at the same time, computer desktops can become more cluttered, with many windows open and visible, and thereby provide ample opportunity for distractions and switches to less relevant tasks. As studies have shown these many open windows and possible distractions can derail focus [1]. Additionally, the more virtually cluttered the computer is by having more applications, windows, or tabs open, the higher the cognitive costs [30] are and the more time workers need to spend to find what they are looking for [20, 37]. Switching between windows and tasks further diminishes our concentration by leaving an “attention residue”, which makes it more difficult
to resume a task once a worker is distracted [23].

In this research we aim to identify ways in which the desktop environment can be leveraged to support strategic multitasking, by providing lightweight guidance that helps information workers maintain focus on relevant tasks, and minimize multi-tasking activities that are distracting and unproductive. To this extent, we designed and developed the WindowDimmer application which predicts the most relevant windows currently in use and dims the other ones to reduce their likelihood of drawing the information worker’s attention.

To test the value of this approach, it was necessary to first establish an understanding of information workers’ practices in interacting with conventional window-based desktops in the context of knowledge work, and specifically to understand their patterns of opening, closing, and switching between windows and tasks. We conducted a formative study to monitor computer based and eye gaze interaction with the window environment of information workers. In addition to yielding surprising findings about the frequency and brevity of task switches, such as the average time of 15.7 seconds between two window switches, this study also provided a baseline against which we could compare participant’s task and window switching practices using WindowDimmer. Our subsequent study of WindowDimmer in use indicated a substantial reduction in brief window switches, and visits to windows not relevant to the current task. Additionally, participant feedback to the application was generally positive, and indicated that WindowDimmer could help reduce distraction and improve focus.

In our work, we focused on one community of information workers, software developers. We studied software developers because of their extensive use of computers at work, their openness towards improving their work and productivity [24], and overall comparable work, while working on a broad variety of activities [32]. However, we believe that although some aspects of the practices we present in this work may be different for other areas of knowledge work, the overall approaches for predicting window relevance and providing desktop-based support for focus may be valuable for supporting information workers in other professions as well.

This thesis makes these primary contributions:

- A monitoring study capturing information on window interaction practices
of professional software developers.

- The WindowDimmer application, capable of predicting the relevance of an open window and applying a dimming effect to the remaining desktop.

- An evaluation of the dimming approach with a second group of professional software developers and software engineering students.
Chapter 2

Related Work

Related work can be roughly categorized into the following areas: studies on understanding information workers’ interactions with their window environment, approaches to detect currently relevant windows, and approaches to support task focused work.

2.1 Window Interaction Studies

Modern window-based desktops allow information workers to access and see many applications and windows simultaneously with many applications capable of opening multiple windows. With increasing screen size, multi-monitor support, and the ability to use several applications at the same time, they also provide sources for distractions, such as multi-tasking between the main work task and less relevant task, thus reducing focus [1]. A high number of open and visible windows creates distracting visual clutter, but windows can introduce a cost even when not visible. Many open windows make it harder to locate the ones needed for a task, increasing the cost of window switches [20]. Each window switch can potentially act as a trigger to decoy the worker to perform a task switch, such as a new unread email, an interesting Twitter conversation, or a not yet finished task. Work by McMains and Kastner [30] and Niemelä and Saariluoma [37] has shown that having more windows or tabs open on the computer at the same time, increases cognitive costs and requires more time for workers to find the relevant window. Little is known
yet about information workers’ practices in interacting with conventional windows-based desktops at work, such as opening, closing and switching between windows and tasks. Existing studies vary in the type of information that was tracked. Mostly, studies either focused on active observations [47] or specific computer interaction events based on keyboard and mouse interaction with specific applications [13, 16]. The interactions of software developers in particular have been studied within their IDEs for the Pharo IDE [34] and Eclipse [36]. Our work is not restricted on an individual application, but considers interactions with all programs and windows on a developers’ computer.

In 2004, Hutchings et al. conducted a more generic monitoring study across all windows to investigate the effect of larger screen real estate on window switching behavior [17]. They found that users with smaller screens had a median of 4 visible windows, while those with larger or more screens had a median of 6 visible windows. With ever increasing screen sizes and resolutions, we expect an even larger number of open and visible windows. Our studies will monitor similar features and provide an update to these numbers.

Further studies that observed information workers used the interaction to determine tasks [32] or higher-level working spheres [15, 25]. These studies were conducted over multiple days and reported a very fragmented working style with many task switches and interruptions. For instance, González and Mark [15] defined units of work as working spheres between which information workers switch frequently. In their study with 24 information workers they found that their work is highly fragmented and switches between working spheres occur on average every 12 minutes Mark et al.

Our first study uses eye-tracking to investigate visual attention on the desktop environment. Eye-tracking has traditionally been used to study reading and comprehension. Early experiments studied the differences in code comprehension between novice and expert developers [5, 8]. Other research used eye-tracking technology to investigate the comprehension of specific software artifacts, namely class diagrams [55], design patterns [45], and identifier styles [46]. In more general contexts eye-tracking was used to measure task difficulty [4, 14], mental workload [18], and cognitive load [22]. We aim to leverage the eye-tracking technology to gather task-independent insights into how software developers visually interact
with the windows on their desktop.

2.2 Detecting Relevance

There is a range of approaches that have tried to detect the relevance of windows [39], artifacts [21], and groups of applications [40, 48]. Applications for using the relevance of related resources include task resumption, task switching, and self-monitoring. Most of these approaches take as basis two types of features; temporal and semantic. Temporal features are related to the order in which resources are accessed. For instance, Bernstein et al. used switches between two windows to calculate how closely-related they are [6]. They used their WindowRank to arrange windows for easier task-related window switching.

Semantic features are usually shared words in the title and content, for example used to group open applications and documents into tasks and provide users easier access to task related resources [11]. Oliver et al. used both types of features to analyse window switches [38] and provide an alternative window switching application [39]. In SWISH [38], they calculated temporal and semantic features to group windows into tasks and achieved an accuracy of 70% for pre-recorded data. RelAltTab [39] also uses both temporal and semantic features to measure window relatedness and reorder or highlight the windows in the window switching application. Their user study showed that user switched to related windows in over 80% of the instances and that reordering windows had a negative effect.

2.3 Supporting Focused Work

The high fragmentation of a software developer’s typical workday is well reported [15, 25, 44]. On average, there are switches between different activities every few minutes even when not all of them are task switches [32]. Times with fewer task switches and higher engagement with tasks are reliably rated as more productive [27, 31]. Successful approaches for increasing the perceived productivity have, for instance, reduced distractions of not work-related websites completely by blocking them [29] or reducing their availability [52] which gave participants a higher feeling of being in control.

Generally multiple windows are required to complete a tasks. By reducing
the time and effort to find the right window when switching users can keep their focus and resume tasks faster. There are several approaches to assist users with window switching [19, 50, 51, 54], but their work focuses solely on providing a better overview and speeding up window switching. Others, more closely related to our work, have tried to reduce the overload that people experience by grouping tasks and documents [11], grouping windows [48], improving access to occluded window content [53], or reducing the visibility of secondary screens [10]. The latter approach by Dostal et al. tested several approaches on how to reduce distraction from a separate, non-focused screen. Dostal et al. examined the proposed approaches in a qualitative study with one participant who perceived a technique that dims the whole screen, but visualises display changes on the pixel level as most useful [10]. Different to our work, the do not consider individual windows but only dim out the whole screen.

Specifically for software developers, approaches have been proposed to reduce the “window plague” in the window-based Pharo IDE use temporal features to highlight the most important windows and automatically close the least important ones [35, 41]. Both of these approaches are restricted to the IDE (a single application) and were only validated based on recorded data, but never tested with users.
Chapter 3

Study 1: Developers’ Window Interactions

To support information workers’ focus at work by emphasizing relevant windows, it was first necessary to establish an understanding of their practices in interacting with conventional windows-based computer desktops. Specifically, we were interested in better understanding developers’ patterns of opening, closing, arranging and switching between windows on their monitor(s). In addition, we collected this data to use it as a baseline for WindowDimmer’s model in study 2.

3.1 Monitoring Study

In this first formative study, Raphael Rosenast, a MSc student at the University of Zurich monitored 18 professional software developers for an average of 17.9 days (±12.9) per participant in their usual work environment. For the period of the study, he ran a monitoring application on participants’ computers in the background, which collected window interaction data, user input data and eye-gazes.

3.1.1 Participants

Raphael recruited 18 professional software developers from three companies of varying size in the software and computer hardware industry through personal contacts. He focused his study on professional software developers—as one commu-
nity of information workers—to make sure that participants largely use the computer for their work and have similar work habits [32, 33]. Also, he limited his study to users of Microsoft Windows as their main operating system due to the compatibility with the eye-tracker and his monitoring application. Of his participants, 17 were male and 1 was female. They had an average of 17.5 years of software development experience, ranging from 2 to 35 and described their main responsibility as development with accompanying tasks of project management, system engineering, and testing. They all resided in Switzerland.

3.1.2 Study Method
Raphael started the study by handing out consent forms and explaining the purpose and process of the study to participants as well as informing them about the monitoring application and the data it collects. He also informed participants that their participation is completely voluntary and that they are allowed to withdraw at any point in time. He then provided instructions and supported them with installing the monitoring application on their computer and with installing and correctly positioning the eye-tracker in front of their primary monitor. He chose the Tobii 4C as the eye-tracker for his study due to its portability and affordability. Note that since it was technically impossible to hook several Tobii 4C’s to the participants computer, he could only capture eye-gazes on the participants’ primary monitor. Once the setup was complete, he asked participants to continue their regular work as usual while the monitoring application and eye-tracker collected data nonintrusively in the background. At the end of the study, he revisited each participant, collected the monitoring data from the participant’s device, uninstalled the software and the eye-tracker, and conducted a short follow-up interview to ask for feedback on the study.

3.1.3 Monitoring Application
To collect computer interaction and eye-tracking data, Raphael developed his own monitoring application that runs in the background based on the PersonalAnalytics

\footnote{https://gaming.tobii.com/product/tobii-eye-tracker-4c}
project\footnote{https://github.com/sealuzh/PersonalAnalytics}. For compatibility reasons with the eye-tracker, he developed his application for the Windows 10 operating system. His application collects data on all mouse movements and keyboard events without tracking the specific keys being pressed for privacy reasons. Mouse movements are recorded as a single event with the time and the end position of the cursor. In addition, the application logs all window events, including focus, create, destroy, move, and resize events together with the size and location of the window, the title of the window and its state (active, minimized or maximized), and information on all other open windows and their position. Finally, the application also collects information on all connected screens and the dimension of each screen. In terms of eye-tracking data, his application captures eye fixations—the location on the screen the user visually pays attention to—including fixation position and duration as provided by the Tobii eye-tracker.

3.1.4 Collected Data

Raphael collected a total of 322 days of data for this study, ranging from one to four weeks worth of data per individual participant. All participants reported the days they were monitored on as regular work and did not report any distraction or changes in their work based on his monitoring application and the eye-tracker. Overall, he collected a total of 9,522,956 window interactions and 63,292,622 eye fixations from all 18 participants.

For each window interaction, we calculated how many pixels of each open window was visible based on the order in which windows appear on the monitor. We further determined how long windows were open or active based on the duration between window events. From the mouse movement data, we derived periods of time when participants were not active on their computer and excluded these from our analysis. In particular, we chose a 5-minute threshold and considered every period without mouse movement that was longer as not active and excluded it.

To identify the windows a participant looked at, we mapped each captured eye fixation to a window using our processed window interaction data that includes the visibility of all windows, their location and size. Overall, we were able to distinctively map 86.6\% of the recorded fixation points to a window. In cases
when a participant looked at the taskbar, a monitor with no windows, or a newly opened window that was not yet registered with the windowing system (which can happen in the first second of opening a new application), we were not able to map the eye fixation to a window.

By comparing and validating the data of all participants, we noticed one outlier, participant P6, that had more than twice the amount of open windows compared to all other participants. We therefore decided to exclude the data of P6 from the further analysis.

3.2 Results

This section presents the results of our quantitative analysis of the logged computer interaction and eye fixation data. While Raphael provided an initial analysis of the collected data, all the results reported in this thesis were newly calculated from the data. We start with an analysis based on developers’ computer interaction (Section 3.2.1 to 3.2.4) and finally present the results from an analysis of the eye-tracking data (Section 3.2.5).

3.2.1 Open Windows Behavior

To better understand the potential for distracting developers when switching windows, we analyzed how many windows and applications developers have open at any given time. We calculated the weighted number of open applications and windows by the duration they were open. Figure 3.1 presents the number of open windows across participants. Overall, developers had an average number of 12.1 (±6.6) windows open at all times from an average of 9.5 (±3.0) applications. Also considering their multi-monitor setup, we observed that single monitor users tend to keep an average of 6.9 (1 participant only), dual monitor users keep 12.6 (±4.0), and triple monitor users keep 14.9 (±11.4) open windows at the same time, which is comparable to previous work [40]. Surprisingly, the overall screen size did not significantly impact the number of open windows (Pearson correlation coefficient of 0.21 with p = 0.42).

We found that the number of open windows varies over the course of a day. In particular, developers opened more windows than they closed, leading to a growing
number of open windows over the course of the workday, from 9.9 (±5.5) in the morning to 14.4 (±7.3) in the evening. While this suggests that users might have turned off their computers in the evening, we do not have this information. Despite the variation in the number of open windows each developer has open, this increase over the course of a day is consistent across all participants as illustrated in Figure 3.2, and similar to the increase in the number of windows open inside the Integrated Development Environment (IDE) that Roethlisberger et al. found [41].

Developers generally do not close windows frequently, but rather leave them open in the background, although, some developers at least tend to reorganize and clean up their desktop a bit more after returning from lunch break or before leaving work. We also found that developers very rarely move or resize windows, compared to the frequency of other window interactions, such as switching, opening and closing. Less than 3% of all window interaction events that were captured over the course of the study were changes of size or location of a window.

3.2.2 Window Switching Behavior

Based on the computer interaction data, on average, each window was open for a total of 79.2 (±60.1) minutes, yet this value varied a lot, ranging from 19 minutes to 4 hours across participants. Nonetheless, we found that developers switch between windows very frequently keeping a window in focus (active) for only very short amounts of time. On average, developers focused on a specific window for only
15.7 (±77.3, median=2.2) seconds, before switching to the next one. The high number of switches are comparable to previous work [15, 32, 33] where researchers looked into activity and task switching which are related to window switching.

As illustrated in Figure 3.3, the short time developers spend in a window is right skewed by a high number of very short window switches. Overall, 32.9% of all window switches lasted less than 1 second and only 30.7% of the windows remained active for longer than 5 seconds before the developer switched away again. The large number of short window switches might occur for several reasons, including the developer navigating through the open windows to find the relevant one, the developer closing irrelevant windows or selecting the wrong one, or from being briefly distracted. Most window switches are switches to previously open windows. In total, developers revisited already open windows in 76.6% of the window switches and only in the remaining 23.4% they opened a new window.

### 3.2.3 Multi-Monitor Usage

Even though 94.1% (16 out of 17 participants) are using a multi-monitor setup, our data shows that developers are using their primary monitor during the majority of the time they spend on the computer and position the most windows on. On the Windows 10 operating system, one monitor is defined as the primary one, for which
the coordinates [0/0] are assigned to the left upper corner, the remaining ones are secondary monitors. The secondary or tertiary monitors were actively\textsuperscript{3} used only during 30.7\% (±23.3\%) of the time, and for 23.9\% (±19.8\%) of the windows, on average. 94.1\% (16 out of 17 participants) were using a multi-monitor setup, 14 (82.4\%) worked with two monitors, the remaining 2 (11.8\%) worked with three. Several developers used a laptop and connected the laptop to an external monitor for most of the time, but sometimes also just used the laptop and therefore only a single monitor configuration. Over the course of the study, we found that 64.7\% (11 participants) switched their monitor configuration from time to time, while 35.3\% (6 participants) consistently used the same monitor setup throughout the study.

### 3.2.4 Usage of Screen Real Estate

We found that developers have several windows open and visible at the same time. Over all monitors, there were always an average of 1.5 (±0.7) windows fully and 3.3 (±1.7) windows partially\textsuperscript{4} visible. The currently active window, which on Windows 10 is always fully visible, thereby took up 84.7\% (±9.7\%) of the monitors’ screen real estate and was maximized\textsuperscript{5} 60.6\% (±48.9\%) of the time.

\textsuperscript{3} An “active window” means it is currently focused and/or receives keyboard input.

\textsuperscript{4} In a partially visible window, another window is partly overlapping, and thus covering, it.

\textsuperscript{5} A maximized window takes up the entire screen real estate of the monitor, other than the taskbar.
3.2.5 Visual Attention and Focus on Windows

The results on developers’ window switching behavior so far are based on developers’ computer interaction. Hence, it is still unclear whether developers are actually looking at the active window, or at another open window that is currently not in focus. Since a better understanding of the visual attention also allows us to better understand which windows are relevant, we equipped participants with eye-trackers on their primary monitor.

The results of our analysis show that visual attention shifts similarly frequently as the active window, with an average of 31.2 (±87.8) seconds per window and a median of 4.7 (±20.4) seconds. The distribution of shifts in visual attention between windows is again heavily skewed, with 19.3% (±8.4%) of the shifts being less than a second long. The longest time we observed a developer looking at the same window was 64 minutes, significantly lower than the 4.7 hours for the active window. This is however not surprising, since developers might just look away in between for short periods to relax their eyes.

Using the captured eye fixations, we further found that the visual attention matches the actively selected window in 83.1% of the cases on the primary monitor. The other 16.9% of the eye fixations were directed at windows that were open and visible, yet not the active one. It is not surprising that developers do not always look at the active window, and in fact, developers looked away from the active window at least once for 67.7% of all active windows. When they did so, they looked at an average of 2.2 (±3.2) distinct non-active windows. There were, however, also 32.3% of the active windows that participants never looked away from until they switched the active window using mouse or keyboard. Overall, this analysis provides evidence that using the active window is a relatively good indicator for a developer’s attention.

3.2.6 Behavior by Activity

While we cannot exactly know what a developer was working on without observing them, the application and title of the active window can be used to estimate the activity. We observed different window interaction behavior depending on the type of activity a window belongs to. Our participants tended to keep some types
of windows open in the background even when not used while others where only opened for a specific task and then closed again. The File Explorer (211.2 ± 110.9 minutes), email clients (167.4 ± 164.0), and chat application (165.4 ± 120.2) windows were open for significantly longer than windows of applications for programming (51.9 ± 40.8) or code review (9.7 ± 15.0).

We have also observed differences in the time between window switches depending on the activity. Email client windows (18.7 ± 28.6) and web browser windows with work related pages (13.3 ± 9.5) are active for significantly longer than all other types of activities which are only active for 3.6 (± 4.8) seconds at a time.

Whether a window of an activity is often maximized varies depending on the activity’s interaction with other windows. On the extremes, with 94.2% (±23.4%) windows for code review—a rather independent activity—are almost always maximized, but File Explorer windows almost never with 0.7% (±4.9%). The File Explorer is mostly used to select and open a file or document to start or continue an ongoing and usually does not require a large window. In between, email client windows and programming windows are maximized 52.5% (±46.4%) and 44.7% (±38.4%) of the time respectively. These activities can require large windows and undivided attention, but often require external resources and references as well.

3.3 Discussion

The results of our first study showed that most developers keep a large number of windows open at once, providing ample opportunity for distractions and switching to a non relevant window [1, 30, 37]. The high frequency of window switches and short duration a selected window stays active further shows the potential of distractions and might be due to several accidental switches that could lead to getting distracted from the primary task.

According to discussions with the participants from study 1 and our own experience, very few tasks require a high number of open windows. Yet, developers are not taking any measures to potentially reduce distractions, which points to the value of a lightweight automated approach to reduce distractions by open windows and support developers in their focused work.

Finally, our results show that using the computer interaction can be a good indi-
cator for determining a developer’s attention. Therefore, we will base our relevance model in the following solely on the computer interaction due to the potential of its applicability and lesser invasiveness when deployed with information workers.
Chapter 4

Predicting Relevant Windows

To develop a lightweight approach that helps information workers maintain focus on the relevant windows and minimizes unintentional switches, we explored the prediction of relevant windows.

4.1 Model

We leveraged the results from study 1 to develop a model on predicting the most relevant windows. We used the following temporal and semantic features that have been applied successfully in previous work [11, 38, 39]. Note that other features that we identified from related work or our own experience would have made the model either too complex or not allow a real-time application in a real-world setting that we implemented in study 2.

4.1.1 Temporal, Recency

Recency is one of the simplest and most commonly used temporal features that scores a window based on how recently it has been active, so that the last window before the current one receives the highest score. To calculate a recency score we focus on the window activation events in the computer interaction event log. Specifically, we start from the current window and go back in time to the ten previous windows, remove duplicates, and give a score in reverse chronological order. From an order of $A \rightarrow B \rightarrow C \rightarrow B \rightarrow D \rightarrow C$ with $C$ being the active window we get
a ranking of $D, B, A$ as the active window is not scored and duplicates are removed.

4.1.2 Temporal, Duration

Our duration feature scores a window by how long it has been active during the last 10 minutes. The score is calculated by dividing the summed active duration by the observation interval, in our case 10 minutes. We picked this interval to provide a good balance between keeping the measure stable from short recent switches and only including windows that are related to the current task. Task switches are reported to occur every 3 to 12 minutes, so the current task should in most cases be entirely covered by our interval [27, 32].

4.1.3 Temporal, Frequency

The frequency feature shares the same observation interval with the duration feature. Windows are scored by counting the number of switches to that window: the more switches to a window, the higher the rank of the window. Many tasks involve switching frequently between the same few windows. This feature is intended to capture that behavior.

4.1.4 Semantic, Window Title

We assume that windows whose title and textual content are similar are also themselves related and therefore relevant to each other. We want to use semantic features to predict how related two windows are. To calculate a score in regard to predicting window switches we are interested in how related a window is to the currently active window. We are using the window titles to determine relatedness. While the contents of a window could provide more information, they are not as easily available as the window titles, which are exposed by operating system interfaces, and reading the contents of all windows is very invasive, introducing justified privacy concerns by users. We processed the window titles by removing all punctuation, filtering out stop words, and stemming the remaining words. The weight of the stemmed words in the processed window titles of all open windows are calculated by the term frequency–inverse document frequency. All processing operations are performed with tm, a library for text mining in R [12]. With these weights we cal-
calculate the cosine similarity of the window title of each open window to the window title of the currently open window. The open windows are ranked by the similarity value.

### 4.2 Empirical Analysis based on Study 1

After calculating the scores for the temporal and semantic features described above, we used the data collected during our monitoring study to determine the weight of each of the features to calculate a combined score. As a starting point, we looked into a linear equation due to its simplicity and speed of calculation. More complex equations and models as a topic of future work are discussed in Chapter 7.

For each window switch we calculated a score for each open window equal to \( \alpha \times \text{recency} + \beta \times \text{duration} + \gamma \times \text{frequency} + \delta \times \text{semantic} \).

We investigated the features individually and created a ranking based on their predictive power. Figure 4.1 shows that the order is not easy to determine and depends on the target number of windows that are predicted. When we consider only our highest scoring window, the feature with the highest predictive power is recency, followed by semantic, duration, and then frequency. When considering the top 2 or top 3 results, the order changes as the semantic feature becomes less predictive.

As a first step we tested all features with equal weight as well as all combi-
nations of 3 out of 4 and 2 out of 4 features. Figure 4.1 shows the performance of the four individual features, a combination using equal weights \((mean)\), and our final weighted combination \((weighted)\). Using equal weights interestingly performs significantly worse than either of our best two features. The best combination we found only consisted of three of our features containing recency, semantic, and duration. Adding frequency reduced the result slightly. We then proceeded to examine various weight schemes of the three taking into account the order. We used a binary search approach with intervals of 0.2, 0.1, and finally 0.05 to gradually optimize the weights. The optimal combination we found that way is \(0.5 \times \text{recency} + 0.45 \times \text{semantic} + 0.05 \times \text{duration}\). While its weight ended up being rather small, the duration feature improves the score by filtering out switches to windows that become active and relevant for a very short time, but should not change the whole context of the task.
Chapter 5

Study 2: Highlighting Relevant Windows to Support Focus

We explored how to visualize the most relevant windows to the user and de-emphasize the remaining ones. To that purpose, we extended the monitoring application with a component, called WindowDimmer, which fades away all windows that are not considered relevant.

5.1 WindowDimmer Approach

For our approach called WindowDimmer, we extended the monitoring application that we developed for study 1 (Section 3.1.3) with the model to predict relevant windows that we determined in the empirical analysis (Section 4.2). For WindowDimmer, we chose to show the top 3 most relevant windows and reduce the visibility of all others. We chose 3 as the threshold as developers tend to not only fixate the active window, but an average of 3.2 distinct windows including the active one (Section 3.2.5). Hence, for WindowDimmer, we predict a ranked list of the 3 most relevant windows using our combined and weighted predictor (weighted in Figure 4.1) which reaches a 72.7% probability of selecting the correct three most relevant windows. Whenever WindowDimmer receives an event that the user switched between windows, we gather a list of the remaining open windows and their window titles, extract their features, and calculate the relevance score.
Whenever a window title changes without a window switch, such as when the user selects a new tab inside the web browser, we recalculate the relevance scores.

As visualized in an example in Figure 5.1, WindowDimmer reduces the visibility of all windows that are not the top three most relevant ones, by dimming them. Concretely, the active window and the two windows with the highest relevance scores remain untouched, while all other remaining windows and the desktop background, in case it is still visible, get slightly dimmed by applying a gray overlay with 25% opacity. The Windows 10 taskbar stays visible and un-dimmed to allow easy navigation between windows. Note that the most relevant windows are not necessarily all in the foreground, since the order of how Windows 10 stacks the windows is based on recency only, while our model includes duration and semantic similarity as well.

As mentioned above, by default WindowDimmer highlights the top 3 most relevant windows (including the currently active one). WindowDimmer provides features to adjust the number of windows that don’t get dimmed and pause or disable the dimming (see Figure 5.2). As with the monitoring application, WindowDimmer is based on and integrated into PersonalAnalytics\(^1\) and as such is restricted to Microsoft Windows 10. The calculation of scores and application of the dimming is based on the background monitoring and occurs instantaneously after receiv-

\(^{1}\)https://github.com/sealuzh/PersonalAnalytics
The calculation of feature scores was reimplemented from the reactive approach in the analysis to run in real time using the Accord.net framework [49] for text processing of the window titles and scoring of the semantic feature.

### 5.2 Intervention Study

To evaluate the potential of increasing focus by dimming irrelevant windows, we conducted a second, two-part field study. In the first part, we investigated if our model can identify the top three relevant windows as reported by participants. In the second part, participants used WindowDimmer in situ during their real-world work and provided us with qualitative feedback.
5.2.1 Procedure

In the beginning of the study, we asked participants to read through the consent form, ask any questions they might have, and sign it. After that, participants were asked to install the monitoring tool with the WindowDimmer extension on their main work computer. Finally, we asked participants to continue their work as usual for the next 5 workdays.

During the first part of study 2, which lasted three of the five days, we logged participants’ interactions with their windows, and asked them to answer a pop-up survey in regular 40-50 minute intervals (to have some randomization). The pop-ups, as seen in Figure 5.3, listed all currently open windows and asked participants to tick a check box of all the windows that are relevant for their current work. No definition for relevance was given, as we did not want to bias participants’ responses by restricting relevance to work-related windows for example. With this part of the study, we wanted to examine whether our predictive model based on user interaction is also able to accurately predict user-defined relevance.

During the second part of study 2, which lasted the remaining two days, we continued logging participants’ computer interaction and enabled WindowDimmer. We disabled the self-reporting pop-up to prevent frustrating participants in case their selection of relevant windows was different to the windows that our approach dimmed. At the end of the second part of study 2, we conducted semi-structured
interviews to receive feedback on their experience with WindowDimmer. We also
asked them to answer a survey with demographic questions as well as the Sys-
tem Usability Scale [7], a standardized survey for evaluating the usability of our
approach. Finally, we collected the logged data from participants’ machines, af-
ter giving them the opportunity to obfuscate the logged data to alleviate potential
privacy concerns. We then uninstalled the monitoring application and gave partic-
ipsants $30 US to compensate for their efforts.

5.2.2 Participants
We recruited a total of 12 software developers; 6 professional software develop-
ers who worked for 4 different software companies in Canada, the US, Germany,
and Switzerland, and 6 computer science students (1 undergraduate, 4 graduate, 1
postdoc) in Canada and Switzerland. Contact to the participants was established
through personal contacts. Participation was entirely voluntary. 5 of our partici-
pants identified as female, the remaining 7 as male. The average age is 26 with an
average of 2.6 years of experience for the working participants.

5.2.3 Collected Data
We collected all data required to assess our approach. We collected data on con-
nected screens, open windows, and window interaction comparable to the moni-
toring study as described in Section 3.1.4. We additionally recorded the calculated
relevance scores for each feature as well as the summed score and rank of the open
windows. For each window in a submission of the pop-up by a participant, we
recorded the response and whether our model predicted the window as relevant or
would have dimmed it. In total we collected 266 pop-up responses with an average
of 22.2 (±9.5) responses per participant over three days. The number of responses
varied between 11 and 36, depending on how much time the participant spent on
their computer. Each pop-up contained a list of 14.5 (±11.8) open windows.

5.3 Results
The intervention study provides similar data to the monitoring study, but extends
it with information related to our model and the WindowDimmer approach. We
compare data on computer desktops, windows, and window interaction with the previous results, investigate participants’ reports on relevant windows, and compare window interaction behavior between the two parts of the study.

5.3.1 Window Interaction Behavior

The participants of the intervention study had similar screen setups as the participants of the previous monitoring study. All participants used a laptop with six participants adding one and three participants adding two external screens. We calculated the number of open windows with the same methodology as before and observed 13.5 (±11.1) open windows across participants, ranging from 4 to 31, which is a slightly higher average compared to the 12.3 open windows in the monitoring study. The growth of open windows over the course of the day, from 10.0 (±8.0) in the morning to 14.3 (±8.9) in the evening, is also comparable, though less stable due to fewer participants and a shorter observation time.

Unlike the over 30% before, only 17.0% of the captured window switches lasted for less than one second, although the distribution is similar to the one in Figure 3.3. This could be due to different applications used by the participants influenced by their different job roles. The proportion of window switches to already open windows is also lower by about 8 percentage points with 67.8%.

5.3.2 Predicting Relevant Windows

As a first step, we wanted to evaluate whether the predictions by our relevance model overlap with the perceptions of the participants. Therefore, we compared the reported relevance of open windows in the first part of the study with the top 3 relevant windows predicted based on user interaction. In the 266 self-reports that we collected participants reported 1.8 (±0.9) windows to be relevant out of the 14.5 (±11.8) windows that were open and listed in our pop-up. This number includes the currently active window for 85.3% of all pop-ups.

When we compare the top 3 windows that our relevance model predicted with the windows reported as relevant by the participants, we found that 88.3% of the predicted windows matched the self-reports (10.4% true positives and 77.9% true negatives). 9.8% of all windows were predicted relevant by our model, but
Figure 5.4: Number of windows reported (by participants) and predicted (by our model) relevant averaged over all reports. The three top scoring windows in our model are considered relevant.

not by the participants (false positives). Only 1.8% should have been considered relevant according to our participants, but were not predicted as relevant based on our model (false negative).

Figure 5.4 shows the relevance results per participant. To account for the varying number of responses per participant, the values are averaged per pop-up. The number of windows incorrectly predicted as not relevant, incorrectly predicted as relevant, and correctly predicted as relevant are very similar across participants. The value that is varying the most is the number of windows that were predicted and reported as not relevant. This is most likely due to the the varying number of open windows per participant.

5.3.3 Evaluation of WindowDimmer

During the study period in which participants used WindowDimmer an average of 30.4% of the computer desktop area (all screens) was dimmed. The results also show that the dimmed area increases with the size of the computer desktop with a Pearson correlation of 0.59 (p=0.00011). Figure 5.6 illustrates this effect. During the study, all of our participants used a laptop with 9 participants usually connecting at least one additional screen (see Section 5.3.1). Their screen resolution ranged from 1280 * 680 to 3840 * 2160.
During the intervention phase with WindowDimmer activated, participants had fewer window switches and spent longer time in relevant windows. The number of window switches lasting less than one second decreased from 18.4% to 15.3%. However a paired t-test showed that this is not significant (p=0.1372). Figure 5.5 displays a breakdown by application type of the length of window activations. While there is a decrease of very short window activations across all types of applications, the ratio of very short window interactions varies. Applications that require a higher level of focus and concentration like IDEs and web browsers stay active for longer times while email clients have more very short activations.

Having WindowDimmer reduce visibility of open windows might encourage participants to open more windows as they are no longer as distracting. However, we do not see a consistent change in the number of open windows across participants. While the number decreased for 6 participants, it increased for the other 6. The number of windows seems to be much more related to the type of work and tasks performed during the day.

For each window switch by a participant we recorded whether our model predicted the target window to be relevant and the rank of the target window. Window switches to windows not predicted to be relevant and therefore dimmed during the second part of the study decreased by 10.2%, from 48.2% to 43.3% with the
Figure 5.6: Percentage of desktop area dimmed by desktop size. Both are calculated across all screens.

dimming active while switches to the window WindowDimmer predicted as most relevant increased from 29.8% to 33.1%. The remaining switches target the second most relevant window, which is the third window not to be dimmed, where we also saw an increase by one percentage point.

5.4 Participant Feedback

Our post-study survey included the System Usability Scale (SUS). We measured a mean SUS score of 72.7 which is considered to represent a “Good Usability” based on a large survey of SUS scores of previous studies [3].

We further interviewed participants after they completed the study to collect qualitative feedback and ask about specific situations, where they perceived our approach to be helpful or hindering to their work. Generally, the WindowDimmer approach was perceived as useful by 8 of the 12 participants. They mentioned the window dimming was “helpful”, or “worked well”. The other 4 participants reported it having no or very little effect, but also not hindering their work.

P9 and P11 reported an interest in dimming everything but the currently active window “to really focus on the current task”. Leaving only one active window undimmed instead of three might help them to stay better focused. Additionally, they liked that the dimming not only applied to other windows, but also the desktop background itself. These two participants found their desktop background
“normally very cluttered and that can be pretty distracting”. With one participant calling himself “not a great organizer of my desktop”, dimming the desktop background reduces the focus on the clutter.

Three participants (P6, P9, and P10) found themselves distracted by sudden changes in the dimming. When they had too many open windows, they wanted to make sure WindowDimmer was not dimming anything important, which caused them to look at windows that just had the dimming applied. P9 and P10 would prefer a smoother transition leading to the full dimming over a few seconds.

All participants generally agreed on the problem of decreased focus when working on their window-based computer desktops. Although outside of the scope of WindowDimmer, many participants reported having trouble finding the relevant content within tabbed interfaces, especially the web browser. Two participants (P2 and P10) suggested applying a similar dimming approach to tabs they hadn’t used in a while and were no longer relevant.
Chapter 6

Threats to Validity

The biggest threats to the validity for our results are external validity threats due to the limited number of participants and the type of information workers that participated. We elaborate those and the threats to construct and internal validity in this section.

6.1 Construct Validity

Both of our studies were conducted in the real world to gather data that is as realistic as possible. Using a monitoring application in this unsupervised scenario bears the risk of causing inaccurate data to be included due to bugs in the implementation or unexpected restrictions of the participants’ device. To mitigate this risk, we built our monitoring application based on an existing application that had been used in previous studies and conducted study test-runs on various machines prior to running both of our studies.

Limitations to the logged data are that the monitoring application can only capture the participants’ interactions with the computer, and not away from it. Also, the eye-tracker did not allow tracking multiple monitors at the same time, which is why our eye-gaze data is limited to the primary screen only.
6.2 Internal Validity

Monitoring participants might implicitly influence participants’ behaviors, since they are feeling observed. In order to mitigate this risk, we constructed our monitoring application in a way such that data can be collected in the background only. Many participants reported after the study that they did forget about the monitoring application shortly after installing it. In study 1, the eye-tracker could have reminded participants of the ongoing study, but no other interaction was required. In the first three days of study 2, the periodic self-reports (pop-up to collect window relevancy data) might have been considered more intrusive, but participants did not state anything with that regards in the post-study interview. To reduce the intrusiveness of the pop-up, we minimized the amount of time required to select the most relevant windows by adding application icons for quick identification and only requiring a single click to select each relevant window. We further allowed participants to skip individual pop-ups to prevent them from submitting inaccurate information in situations where they were unable to spend enough time selecting the actually relevant windows. In the second part of study 2, where we applied WindowDimmer, we actively influenced participants’ window switching behavior, but this was the intention of this intervention.

6.3 External Validity

Our selection of participants and the total number of participants for either study could limit the generalizability of our findings. While all participants were information workers, participants in study 1 were professional software developers, 6 (of the 12) participants in study 2 were computer science students. We believe that recruiting software developers as a type of information worker for the first study is a good starting point, also to be able to compare better between individuals. Overall, we further tried to mitigate the threat to generalizability by recruiting participants from different companies, countries, and contexts. However, further research is needed to extend this to a broader set of information workers. The daily activities of a developer vary greatly depending on their team, project, and possibly day of the week. We tried to mitigate this by recruiting participants from different companies and staggering the start of the study.
The architecture of our monitoring application required us to restrict the study to participants using Microsoft Windows 10 as operating system. Further studies with extended tooling are required to assess the effect the window managers of different operating systems have on the window interactions and focus of their users.
Chapter 7

Discussion

The objective of our research is to reduce the visual clutter and distractions caused by open windows and thereby foster focused work. To this end, we studied information workers window interactions and developed the WindowDimmer approach to actively dim possibly distracting windows. In a pilot study, we collected valuable feedback from users to smoothen the study process and eliminate bugs in the approach. In the actual study, we then found that participants thought that WindowDimmer was usable and helpful.

We addressed the open and visible windows, but there are obviously many other forms of distractions on the desktop. For example, minimized or occluded windows can be distracting as well, but are not addressed by our dimming approach. Not all forms of distraction effect users the same way. Influenced by the number of screens available and their screen setup, our participants used different layouts of windows which has an impact on how easy it is to block out distractions and focus on the active window.

Addressing Visual Clutter  Similar to previous studies [17], we have seen the size of monitors and number of windows increase. This trend is most likely to continue, increasing the potential for visual clutter from open windows. Predicting the windows that are relevant at any given point in time can be helpful for multiple approaches. We have focused on increasing the visibility of relevant windows as a simple yet effective first step, but can imagine approaches with a larger impact like
minimizing or hiding windows that are not relevant and grouping related windows that are related to a task to have faster access.

Relevance of Tabs Within Windows At this point, we used a granularity level of application windows, yet given the high usage of tabs within windows—especially the ever more important web browser—predicting relevance on a finer granularity would help to provide better support and could have been a useful feature in WindowDimmer. Two participants specifically mentioned the support for tabs in their interviews as they often have up to 100 tabs open in the Chrome browser. The current WindowDimmer approach is not directly transferable to tabs, as there is by definition only one tab visible (per window). Approaches like the Plague Doctor [35] that use color to show relevance could be applied to the tab titles. Alternatively showing a list of the top 3 most relevant tabs on the side could provide easier access for users with lots of open tabs.

Models to Predict Relevance Our model for predicting relevance uses three temporal and semantic features and a linear combination with heuristic weighting of these features. Building on a larger dataset, adding more features, and using machine learning techniques could improve our model significantly. Our studies revealed large differences in the window interaction behavior between our participants. To achieve the best relevance prediction, a personalized model might be required. Some participants wished for the functionality of overriding the relevance by manually setting the dimming of a window. A more sophisticated, personal model could utilize these inputs.

Interactive Approaches Our approach features a model that runs in the background and passively collects user data automatically. To adapt to the various activities a developer engages in over the course of the workday, a more interactive approach could take user settings into account. A very specialized activity like programming could feature a lower number of not dimmed windows and a more change-resistant model, whereas a broader research activity could dim fewer windows and value the semantic relatedness of windows higher. Even more direct
control over dimmed windows could give users the ability to toggle whether a specific window is dimmed or manually mark windows relevant or not relevant for the current task. The individual labels could further be used to dynamically improve the relevance model.

**Accuracy of Eye Tracking**  The eye tracker used in our monitoring study is limited to a single screen. We instructed our participants to configure it for their primary screen which is used most often. Since eye-tracking is becoming less invasive—the model used in our study is a small bar that is mounted on the bottom bezel of the screen—we will most likely be able to capture more accurate data from multiple screens in the near future, which will provide new means for window interactions. In most cases the attention is on the active window, however our participants often looked away from the active window at least once showing us that more than just the active window is relevant.

**Supporting Different Operating Systems**  The current computer market is almost entirely divided between Microsoft Windows, Apple macOS, and the various flavors of Linux (with many different desktop managers). While almost all of those are window-based, the features for managing windows are not the same. Software developers that use macOS were excluded from the studies, but expressed great interest in the dimming approach. Some reported working with mostly non-maximized windows and struggling with the number of open windows. While these studies were for the Microsoft Windows 10 operating system exclusively, it might be worth testing the effect on other operating systems that provide different window management functionalities.
Chapter 8

Conclusion

With the constant improvements in hardware and software technologies, screen sizes and resolution are increasing and usage behavior for window-based desktops is changing. Our monitoring study provides recent insights into how software developers setup their desktop environment and how they interact with windows on the desktop. We observed their work day to be very flexible with a changing number of screens, many open windows, and very frequent window switches. This mirrors the assessments of a fragmented work day found in previous work.

Based on the collected data, we devised a model to predict the relevance of open windows that showed an accuracy of 72.7% when predicting the 3 most relevant windows. Evaluating the model in a second study with a different set of information workers showed that the model can predict relevance self-reported by participants for 88.3% of the windows.

Using the model, we developed an approach called WindowDimmer that reduces the visibility of windows that we predict to be less relevant. The results of our second study in which participants were asked to report which windows they consider relevant and test the dimming approach showed a reduction in window switches and an increase in switches to relevant windows when the dimming is active. 75% of our participants felt that WindowDimmer was useful in supporting their focus on relevant windows. We plan to have a higher impact by improving our model and testing different visualization techniques.
Bibliography


40


