

Master Thesis

20.08.2020

Automated Tab Organization

Roland Schläfli

of Derendingen SO, Schweiz (12-932-398)

supervised by

Prof. Dr. Thomas Fritz

in collaboration with

Prof. Reid Holmes, PhD



University of
Zurich^{UZH}

s.e.a.l.
software evolution & architecture lab

Master Thesis

Automated Tab Organization

Roland Schläfli



University of
Zurich UZH

s.e.a.l.
software evolution & architecture lab

Master Thesis

Author: Roland Schläfli, rolandschlaefli@gmail.com

URL: <https://github.com/rschlaefli/automated-tab-organization>

Project period: 01.03.2020 - 20.08.2020

Software Evolution & Architecture Lab
Department of Informatics, University of Zurich

Acknowledgements

First of all, I would like to thank Prof. Thomas Fritz for the opportunity to work on such an interesting problem, as well as for his great advice throughout the process. Further, I am very grateful to Prof. Reid Holmes for his great inputs, feedbacks, and testing of our approach. Their combined guidance was of tremendous help in the creation of this work. Many thanks also to all of the participants of our user study for their time and engagement. This work would not have been possible without them. Finally, I would like to thank my family and friends for their ongoing support throughout this project.

Abstract

Modern knowledge work is increasingly reliant on online resources. Web browsers are, however, not optimized for the task-based workflows and information gathering needs of knowledge workers. We propose an approach that supports users in curating groups of browser tabs that belong to a task, helping them with task switching and resumption. Furthermore, based on users' browser behavior, we automatically derive such groups and propose them for addition. An *in situ* user study showed that users like the approach and would continue using such a system, but that there are challenges ahead in providing relevant and reusable suggestions.

Zusammenfassung

Modernes Arbeiten basiert zunehmend auf Online-Ressourcen. Web-Browser sind jedoch nicht für die aufgabenbasierten Arbeitsabläufe von Wissensarbeitern optimiert. Wir schlagen einen Ansatz vor, der die Benutzer beim Erstellen von Gruppen von Browser-Tabs unterstützt, die zu einer Aufgabe gehören, und ihnen dadurch beim Aufgabenwechsel und bei der Wiederaufnahme der Arbeit hilft. Darüber hinaus leiten wir mittels des Browser-Verhaltens der Benutzer automatisch Gruppen ab und schlagen sie als Ergänzung vor. Eine Benutzerstudie zeigte, dass den Benutzern der Ansatz gefällt und sie ein solches System weiterhin verwenden würden. Es gibt aber noch Herausforderungen bei der Bereitstellung relevanter und wiederverwendbarer Vorschläge.

Contents

1	Introduction	1
2	Related Work	3
2.1	Artifact Grouping	3
2.1.1	Semi-Automatic Grouping	4
2.1.2	Automatic Grouping	4
2.2	Browser-Based Knowledge Work	6
3	Approach	9
3.1	Tab Grouping Interface	9
3.2	Automated Tab Grouping	11
3.2.1	Monitoring	11
3.2.2	Graph Construction	12
3.2.3	Community Detection	15
3.2.4	Scoring and Ranking	16
3.2.5	Name Generation	17
3.3	User Feedback	17
3.4	Prototype	18
4	Study Method	19
4.1	Procedures	19
4.2	Participants	20
4.3	Data Collection	20
4.4	Data Analysis	21
5	Results	23
5.1	RQ1: Can grouping browser tabs support knowledge work?	23
5.1.1	General Knowledge Work	23
5.1.2	Browser-Based Knowledge Work	24
5.1.3	Challenges in Browser-Based Knowledge Work	26
5.1.4	Tab Grouping in Principle	27
5.1.5	Adaptation of Tab Grouping	28
5.2	RQ2a: Can we accurately group browser tabs that are related based on knowledge workers' workflows?	31
5.3	RQ2b: Can we use our identified tab groups to support knowledge work in the web browser?	32
5.4	Overall Experience and Impact	35
5.4.1	Work with the Extension	35

5.4.2	System Usability	37
5.4.3	Auxiliary Features ("Decluttering")	38
6	Threats and Limitations	41
7	Discussion	43
7.1	Tab Grouping Interface	43
7.2	Automated Tab Grouping	45
8	Conclusion	47
Appendices		53
A.	Architecture	55
B.	Extension: Tutorial	57
C.	Extension: Feature Documentation	61
D.	Interviews: Guiding Questions	67
E.	Survey: Questions	71
F.	User Study: Welcome Document	77

List of Figures

3.1	Tab Grouping Interface: Overview. The overview of our tab grouping interface displays the current tab state of the browser in terms of tabs that are currently open and tabs that have been closed recently. Furthermore, the interface allows for the creation of manual tab groups and enables subsequent interactions with these groups. When enabled, the interface further enables interactions with suggested tab groups. <i>Source: Own depiction.</i>	10
3.2	Tab Grouping Interface: Sidebar. The sidebar view of our interface offers the same functionality as the large overview but can be opened on any site that the user is currently working on. To toggle the sidebar, users can use a dedicated toggle button or use a customizable keyboard shortcut. <i>Source: Own depiction.</i>	10
3.3	Overview of the Automated Tab Grouping Process. The overview visualizes the most important steps of our automated tab grouping approach. <i>Source: Own depiction.</i>	11
3.4	Graph of Tab Switches for our Synthetic Example. Each vertex (i.e., circle) is representative of a tab that has been visited at some point. Edges between vertices represent the switches between the respective tabs, irrespective of their direction. The occurrence count for each switch is marked along the edge. <i>Source: Own depiction.</i>	12
3.5	Reweighted Tab Switch Graph for our Synthetic Example. Each vertex (i.e., circle) is representative of a tab that has been visited at some point. Edges between vertices represent the switches between the respective tabs, irrespective of their direction. The updated weight for each switch is marked along the edge. <i>Source: Own depiction.</i>	14
3.6	Filtered Tab Switch Graph for our Synthetic Example. Filtering has been applied in our default parametrization with a minimal edge weight of two. <i>Source: Own depiction.</i>	14
3.7	Exemplary Tab Group Suggestion. A suggested group as shown in the tab grouping interface. Users could interact with such a suggestion by saving it to their list of curated groups, by discarding the group or single tabs thereof, or by opening the group or any of its tabs. <i>Source: Own depiction.</i>	17
3.8	Feedback Dialog shown on Suggestion Discards. For each group discarded by users, we asked them to rate the suggestion on a scale of 1 to 5, as well as to provide a reason for the discard. <i>Source: Own depiction.</i>	18
5.1	Follow-Up Survey Question D1: "When I am working". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	24
5.2	Number of Open Tabs plotted against Number of Open Windows for each Participant. <i>Source: Own depiction based on the interaction logs of our six primary participants.</i>	26
5.3	Follow-Up Survey Question D2: "The concept of grouping tabs in general". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	27
5.4	Development of Curated Groups over Time. The number of curated tab groups comprises both manually created groups and suggested groups that have been accepted. <i>Source: Own depiction based on the interaction logs of our six primary participants.</i>	29
5.5	Development of Grouped Open Tabs over Time. <i>Source: Own depiction based on the interaction logs of our six primary participants.</i>	29
5.6	Open Tabs and Percentage of Grouped Open Tabs for each Participant. <i>Source: Own depiction based on the interaction logs of our six primary participants.</i>	30
5.7	Follow-Up Survey Question E1: "The automated group suggestions". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	31

5.8	Example for a Wrongly Grouped Suggestion. The suggestion contains artifacts from multiple contexts, i.e., a <i>thesis writing</i> and a <i>release monitoring</i> context. The depicted grouping probably occurred because the tasks were worked on simultaneously, introducing switches between artifacts that do not belong together. <i>Source: Own depiction of a real example observed by researchers.</i>	34
5.9	Example for a Suggestion representing a One-Time Search Task. The suggestion contains artifacts related to a search task for a very specific problem. While it is not wrongly grouped, it would probably not be needed again after the problem is solved. However, some participants found this useful in terms of having a historical view of their past problems. <i>Source: Own depiction of a real example observed by researchers.</i>	34
5.10	Follow-Up Survey Question D3: "When using the extension". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	35
5.11	Follow-Up Survey Question F1: "How useful would you judge the following existing extension features". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	36
5.12	Follow-Up Survey Question G1: "How important would you rate the following potential extension features". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	36
5.13	Follow-Up Survey Question B1: "Concerning the usability of the extension in general" (Positive Statements). <i>Source: Own depiction based on responses as given by our six primary participants.</i>	37
5.14	Follow-Up Survey Question B1: "Concerning the usability of the extension in general" (Negative Statements). <i>Source: Own depiction based on responses as given by our six primary participants.</i>	37
5.15	Follow-Up Survey Question B1: Normalized SUS Scores for all Participants. <i>Source: Own depiction based on responses as given by our six primary participants.</i>	38
5.16	Development of Open Tabs and Windows over Time. <i>Source: Own depiction based on the interaction logs of our six primary participants.</i>	38
5.17	Follow-Up Survey Question C2: "When using the extension". <i>Source: Own depiction based on responses as given by our six primary participants.</i>	39
1	Step 1 of the Initial Tutorial. The first step of the tutorial contains a short repetition of the study procedures. <i>Source: Screenshot of the tutorial embedded in the browser extension.</i>	58
2	Step 2 of the Initial Tutorial. The second step of the tutorial guides users through the installation process for the heuristics engine. Depending on their browser and operating system, appropriate step-by-step instructions will be shown. <i>Source: Screenshot of the tutorial embedded in the browser extension.</i>	59
3	Step 3 of the Initial Tutorial. The third step of the tutorial shortly repeats the most critical information regarding data collection. <i>Source: Screenshot of the tutorial embedded in the browser extension.</i>	60

List of Tables

5.1	Occurrence Count of Task Keywords: Total Count and Unique Count. Total count represents how often a keyword was mentioned overall, while unique count shows by how many participants a keyword was mentioned. <i>Source: Keyword extraction from our interviews including pilot participants (left) and excluding pilot participants (right).</i> 25
-----	---

5.2	Overview of User Interactions with Tab Groups. Includes interactions with both manually created groups and curated suggestions. <i>Source: Statistical analysis based on the interaction logs of our six primary participants.</i>	31
5.3	Overview of User Interactions with Suggestions. Accepted Groups are Suggestions that have been saved in entirety, while Accepted Tabs are single tabs that have been dragged from suggestions to a curated group. Similarly, Discarded Groups are suggestions that have been fully discarded, while Discarded Tabs represent single tabs that have been removed. <i>Source: Statistical analysis based on the interaction logs of our six primary participants.</i>	33
5.4	Overview of Reasons provided when Discarding Suggestions. <i>Source: Statistical analysis based on the interaction logs of our six primary participants.</i>	33
5.5	Overview of Ratings given for Discarded Suggestions. <i>Source: Statistical analysis based on the interaction logs of our six primary participants.</i>	33

Abbreviations

- CAAD** Context-Aware Activity Display
- FDA** Frequency-Duration-Age
- HCI** Human Computer Interaction
- IDE** Integrated Development Environment
- JVM** Java Virtual Machine
- RQ** Research Question
- SUS** System Usability Scale
- UI** User Interface
- URL** Uniform Resource Locator

Chapter 1

Introduction

Web services form a critical part of most technology-supported tasks and, therefore, a lot of the work performed by modern knowledge workers (e.g., software engineers and scientists) tends to happen within a browser environment. However, while knowledge work requires significant focus, cognitive effort, and the gathering and consolidation of information (Dragunov et al., 2005), current browsers are not optimally suited to support such work. Web browsers can quickly become highly cluttered and hard to efficiently navigate (Huang and White, 2010), which might lead to a regular loss of focus by accidental switches to tabs that were not desired. Furthermore, knowledge workers tend to work on multiple tasks in-parallel and a task is often spread across multiple work sessions, making it essential to be able to resume tasks where they are left off (Mark et al., 2005). However, there is little support for such a task-centered way of working in current web browsers (Rajamanickam et al., 2010).

Research has tried to address said challenges by allowing workers to organize artifacts by a task or related search activities, or by applying artifact grouping approaches to automatically do so. However, many of the general artifact grouping approaches are not as applicable to browser-based work, as they focus on broadly grouping all applications used by a knowledge worker and do not integrate as well with specific browser functionality. While there are also relatively few approaches that have been specifically built to support browser-based work, some of these require the usage of a custom browser environment or have other significant restrictions. Furthermore, some approaches are focused solely on search tasks (i.e., grouping artifacts originating from search), whereas task resumption is a problem that occurs in any task performed by knowledge workers. We provide a more detailed overview on the topics of artifact grouping and supporting browser-based work in general in Chapter 2.

In this thesis work, we propose a novel approach for grouping artifacts in a browser environment (i.e., browser tabs)¹. Our approach is centered on two main research questions: the first research question (*RQ1*) is based on the idea of grouping browser tabs in general, and whether that idea can be useful in terms of supporting knowledge work. We formulate this research question as follows:

RQ1: Can grouping browser tabs support knowledge work?

Building on the foundational assumption that tab grouping is a worthwhile endeavor, our second research question (*RQ2*) seeks to evaluate the idea of grouping browser tabs automatically. On the one hand, we evaluate whether we can achieve automated tab groups with reasonable accuracy (*RQ2a*). On the other hand, we evaluate whether such automatically generated groups can be applied to improve the practical workflows of knowledge workers (*RQ2b*). We further assume that *RQ2b* is somewhat dependent on a positive outcome for *RQ2a*, as suggestions that are not at least

¹Hereafter, we refer to browser tabs and browser artifacts interchangeably.

reasonably accurate would not provide much support (or rather, might even confuse or distract users). We formulate *RQ2* as follows:

RQ2: Can we automatically group browser tabs to support knowledge work?

- a: Can we accurately group browser tabs that are related based on knowledge workers' workflows?
- b: Can we use our identified tab groups to support knowledge work in the web browser?

Our proposed approach consists of three parts: firstly, we monitor browser interactions that users perform during work; secondly, we support users in their tasks with tools allowing for the manual creation of, and interaction with, tab groups (*RQ1*); and finally, we apply newly developed heuristics based on community detection to support users with automated group suggestions (*RQ2*). We developed a prototype implementation that can be installed in most modern browsers (i.e., Chrome, Firefox, and Microsoft Edge) and on all major operating systems. Chapter 3 provides more in-depth information on our approach and how the two parts integrate.

To evaluate the applicability of our approach, we performed a two-week user study in which participants performed their everyday tasks while supported by the extension. During the first week of the study, the extension collected relevant data about user interactions with tabs (e.g., how often and when tabs are opened), allowing our approach to determine a baseline model of each user's browsing behavior. In the second week, users additionally interacted with manual grouping features and suggested groups. We performed intermediate (i.e., after week one) and follow-up interviews with each of our participants and additionally surveyed them on their general experience. These responses were then combined with collected data to answer our research questions. We introduce our study method in more detail in Chapter 4.

In our evaluation, we found that the participants of our study often had issues in maintaining a clean browser environment. Participants' usage of artifact management features of browsers (e.g., bookmarks) varied significantly, ranging from very deliberate approaches to using the features only very rarely. Overall, all participants could imagine tab grouping as helpful support in working with their tasks. Some participants did not use our approach as expected (i.e., to map their tasks and switch between them) and instead found novel use cases that we had not anticipated (e.g., using suggestions as a timeline of past tasks). Furthermore, we found that, overall, the suggestions we provided were mostly accurate in describing browsing activities. However, most suggestions were not directly useful, as they represented one-time tasks that would not be repeated. A more detailed overview of our results is provided in Chapter 5, and Chapter 6 presents the most important limitations of these results.

Synthesizing our approach and evaluation results in Chapter 7, we provide an overview of our overall interpretation and key learnings and explain how we imagine further research based on our approach. Concluding our work in Chapter 8, we summarize our overall findings and reflect on the results of our user study and subsequent evaluation.

The **main contributions** of this work consist of:

- A **tab grouping interface** enabling more fluid interactions with browser artifacts, implemented as a cross-platform browser-extension for major browsers.
- An **approach to automatically grouping browser artifacts** using a state-of-the-art community detection algorithm.
- The results of a **two-week user study** with six software engineers based on interviews, survey questions, and statistical measures regarding general knowledge work, browser-specific issues, and the application and potential of tab groups and automated suggestions.
- An overview of the challenges in trying to support task-focused work in the browser, as well as the **potential for future research** regarding the same.

Chapter 2

Related Work

Modern knowledge work is highly fragmented by task switches, interruptions, and other distractions. Research has shown that knowledge workers switch their focus at least hourly (Mark et al., 2005; Sahm and Maalej, 2010; Pilzer et al., 2020). This leads to work being fragmented into many short sessions, whereas workers themselves state that longer work sessions are more productive overall (Meyer et al., 2014; Mark et al., 2015; Pilzer et al., 2020). One part of the problem is that each switch between tasks is associated with a cognitive cost for resuming the switched-to task (Mark et al., 2005; Sahm and Maalej, 2010). Furthermore, interruptions have been shown to incur a resumption cost even when tasks are not switched, as people forget parts of what they had done on the interrupted task (O’Conaill and Frohlich, 1995; Morris et al., 2008). Interruptions can also be more time-extensive: a task could be interrupted and resumed days later, requiring knowledge workers to restore and potentially (re-)find all the artifacts that they had worked with (MacKay and Watters, 2008). Research has explored the area from different angles, like blocking distractions before they occur (Li, 2016; Pilzer et al., 2020), or improving the efficiency of task switches and resumption, whereas we locate our work within the latter area. In the following, we shortly introduce some approaches that sought to support task switching and resumption by grouping artifacts (Section 2.1), as well as the general challenges and a few approaches specific to browser-based work (Section 2.2).

2.1 Artifact Grouping

There has been a lot of research on grouping work artifacts belonging to a similar task (further referred to as “artifact grouping”) and detecting what task a user is currently working on (further referred to as “task assignment”). We locate our work within the same research area but view it as a more specific sub-problem focused on browser-based knowledge work. The idea of artifact grouping is founded on the principle of bundling digital artifacts (e.g., application windows, code source files, or browser tabs) into “working sets” to support task switching and resumption (Henderson and Card, 1986; Oliver et al., 2006). Previous research has addressed the problem in various ways, ranging from having users define their current task manually and detecting the artifacts worked on for that task, to automatically detecting the current task and corresponding set of artifacts. Furthermore, research on artifact grouping has repeatedly included semantic heuristics to evaluate similarities between artifacts or temporal heuristics to derive relationships between artifacts. More recent work often includes these heuristics in sophisticated ensembles to improve overall accuracy. In the following, we provide a short overview of approaches based on manual task assignment (Section 2.1.1) and automated variants thereof (Section 2.1.2).

2.1.1 Semi-Automatic Grouping

Initial research on artifact grouping was largely semi-automatic in that it was dependent on manual task assignment, meaning that the user needed to explicitly state their current task, as well as change it when they switched tasks. Dragunov et al. (2005) introduced TaskTracer, a system to assist knowledge workers with their everyday multitasking. TaskTracer monitors user interactions and builds task contexts (e.g., sets of related artifacts like documents and websites) based on user-defined tasks and manually indicated task switches. Additionally, TaskTracer provides a set of user interfaces that integrate with the operating system and allow users to view, edit, and resume tasks (Dragunov et al., 2005).

Kersten and Murphy (2006) proposed a similar approach with a more specialized use case. Their approach, Mylar¹, supports developers in programming tasks by exposing the internal structure of modular programs, which is not appropriately supported by the primarily hierarchical organization in IDEs. Additionally, Mylar reduces the amount of information displayed by the IDE so that only relevant parts are shown. Based on manual task assignment and the user interaction history, the system learns a structural relationship (i.e., a graph) between the modules of a program. It can then suggest related modules to the user when any of the related files are changed, such that the users can more quickly distinguish files that are potentially affected by a change. Based on the evaluation of their approach in a field study, Kersten and Murphy (2006) found that programmer productivity improved significantly when evaluated based on a user's edit ratio (i.e., the number of edits vs. other interactions with a file) (Kersten and Murphy, 2006).

2.1.2 Automatic Grouping

While tools providing manual task assignment have already been shown to improve productivity, they incur an overhead with regard to maintaining task representations and consistently updating the currently active task. To reduce this manual overhead, follow-up research has explored the possibility of automatically detecting the task that a user is working on, as well as further using this automated task assignment to construct the task context.

Stumpf (2005) extended the TaskTracer system (Dragunov et al., 2005) with two key components: TaskPredictor adds automated detection of the current task (i.e., automated task assignment), and FolderPredictor displays groups of artifacts that are related to the task that is currently being worked on. In their preliminary evaluation, Stumpf (2005) identified several challenges in automated task assignment: users tend to not give many positive examples explicitly, instead requiring an implicit derivation of said examples; users often can only specify what task is not being worked on, as there can be uncertainties about the definition of their current task; to improve the comprehensibility of predictions, systems need to be transparent and provide reasoning; and finally, predictions need to be exceptionally accurate if the user does not have explicit control. In terms of the final challenge, Stumpf (2005) proposed providing the user with suggestions instead of a single prediction, trading some accuracy requirements for additional selection overhead (Stumpf, 2005). As a further development of TaskTracer in terms of automated grouping, Shen et al. (2009) presented TaskPredictor2, an update with improved task prediction capabilities. TaskPredictor2 incorporates a larger set of contextual features and works more efficiently than the previous approach (i.e., in real-time). Overall evaluation with two users showed that the task predictions made by their system were accurate in that predictions were mostly correct or at least reasonable, meaning that the predicted task corresponded to or was related to the current task (Shen et al., 2009).

Oliver et al. (2006) introduced a separate approach to the automated task assignment problem. Their Swish prototype applies semantic and temporal heuristics to a stream of user activity and

¹Mylar has subsequently been renamed to Mylyn and is still provided as an Eclipse plugin at the time of writing.

detects the current task and related windows in a completely unsupervised fashion. The semantic heuristics in Swish are based on the assumption that related windows share some features, which allows for clustering windows based on similarities in their content and metadata (e.g., window titles). As additional support in the decision of which windows belong together, the temporal heuristics construct a graph of window switches and identify the related windows by computing the cliques in that graph. When evaluating their approach with data collected through experiments, Oliver et al. (2006) found that their approach would assign windows to the correct task in over 70% of cases (Oliver et al., 2006). In a further application of their Swish prototype, Oliver et al. (2008) introduced a new approach to window switching with the ALT+TAB keyboard shortcut, the canonical way of switching windows with visual support. Upon invoking ALT+TAB, their approach presents the user with the most relevant windows related to their current task. To evaluate their new system, Oliver et al. (2008) performed a laboratory study, in which they had participants perform a set of tasks and measured their execution time. Overall, their evaluation showed a significantly positive effect when compared to the native ALT+TAB support system (Oliver et al., 2008).

Rattenbury and Canny (2007) introduced the Context-Aware Activity Display (CAAD), an approach to automatically determine and visualize contexts of activities, which they define as “long-term structures whose stability derives from their motivating objective”. CAAD was built to minimize manual user interactions and maintenance overhead, attempting to solve the challenges surfaced in semi-automatic systems. The system is based on a pattern-matching algorithm that derives task contexts from user interaction logs and visualizes these contexts employing a task awareness display. Additionally, the system derives the current task of the user based on its internal representation of task contexts, allowing it to highlight relevant (or hide irrelevant) information in the groupings that are displayed. Furthermore, contrary to preceding approaches, the algorithm used in CAAD allows activities and tasks to evolve. When evaluating their approach in a field study, Rattenbury and Canny (2007) found that the system is generally regarded as useful and provides a reduction in overhead when compared to manual task management. However, participants of the study noted that groupings were not always useful, as they mirror past tasks and cannot be reused for a current or future task (Rattenbury and Canny, 2007).

Bernstein et al. (2008) proposed an approach to artifact grouping and visualization that explicitly incorporates the evolution of tasks, and visualizes artifacts without strict group structures. Their system constructs a graph of switches between windows and computes relationships between windows based on their importance and the switches between them. To visualize groups, the approach relies on the spatial perception of users and maps artifact groups on a canvas without clear group structures, implicitly embedding the grouping in distances and the importance of windows in their size. When evaluating their system in a field study, Bernstein et al. (2008) found that users liked using the system for task-based work but were not completely satisfied with the groupings that were produced. For example, tasks that were unrelated but worked on concurrently tended to be placed close together, which participants saw as inaccurate (Bernstein et al., 2008).

Sahm and Maalej (2010) introduced an approach to artifact recommendation and context switching specific to the developer workflow. Their approach, Switch!, analyzes user interactions with artifacts and recommends the artifacts that will most likely be needed next. To improve the recommendations, Switch! applies a novel method of sharing common data (e.g., projects of a team) with coworkers (Sahm and Maalej, 2010). In further research, Maalej et al. (2017) investigated whether the relationships between two tasks can be predicted using the sets of artifacts that represent each task (i.e., the task contexts). Predicting relationships between tasks could be useful in recommending the next upcoming tasks and could improve task switching in general. Their approach is based on a heuristic they call FDA, which stands for Frequency, Duration, Age, its main components. Using FDA, Maalej et al. (2017) compute the relevance of each artifact regarding its specific task. They then predict the similarity between tasks by computing their overlap, weighting

the contained artifacts with their relevance. In empirical evaluations, Maalej et al. (2017) found that their FDA-based approach works similarly well or better than computing the similarity of textual task descriptions, which the heuristic approach does not even need access to (Maalej et al., 2017).

2.2 Browser-Based Knowledge Work

To the best of our knowledge, the specific problem of grouping artifacts in browsers (further referred to as tab grouping) is not yet thoroughly explored, especially when incorporating automated grouping without manual task assignment. However, there has been research focused on browser-based work and its challenges in general, as well as approaches that tried to address similar problems to our work. Some of these approaches replaced the browser environment with alternative systems that facilitate artifact management and task-based workflows, while others integrate with existing browsers and attempt to support knowledge work and multitasking. In this section, we introduce the challenges in browser-based work as identified by related work, as well as a selection of existing approaches.

The primary artifact in common browser environments, tabs, are powerful abstractions for parallel work and multitasking. However, parallel browsing with tabs that are not properly maintained can quickly lead to cognitive overload (Hahn et al., 2018). The majority of website interactions are revisits of a previously opened page (Cockburn and McKenzie, 2001; Morris et al., 2008). Surveys have shown that many users do nothing regarding remembering artifacts that might be reused (Jones et al., 2003; Morris et al., 2008), and often repeat a search for a website to revisit, instead of using built-in browser tools. However, these users regularly have difficulties in executing the same queries as used previously (Aula et al., 2005; Morris et al., 2008). Furthermore, while browser tools were built assuming that activities and artifacts would be transient (i.e., not related to each other), browsing often happens as part of an overarching task or context (Morris et al., 2008; Hahn et al., 2018).

MacKay and Watters (2008) analyzed the behavior of knowledge workers in tasks that span more than one browser session. While they state that such tasks are frequent, browser tools have not yet evolved to support these multi-session tasks appropriately with the only tools available being bookmarks and browser history. In their study, MacKay and Watters (2008) found that some participants explicitly did not use bookmarks for multi-session tasks, as they considered bookmarks to be too permanent or did not want to clutter their bookmark bar. Some participants approached the re-finding of past artifacts by typing the same search query or parts of the URL that would be autocompleted by the browser based on its history or bookmarks. Furthermore, some participants even used external tools to organize their artifacts, e.g., by sending themselves an email or pasting links into a document (MacKay and Watters, 2008). In further research based on these results, MacKay and Watters (2009) designed and evaluated a set of prototypes for multi-session web tasks, and found that participants searched less and did not use bookmarks as often when being supported with resumption functionalities (MacKay and Watters, 2009).

Morris et al. (2008) proposed a search-focused approach that persists query histories and other actions relating to search activities, allowing users to resume their search across browser sessions. Their approach, SearchBar, records all browsing activities that happen after a search query and organizes these activities in the context of the initial search. Users can manually assign queries to a topic, allowing them to group queries in a similar context. Furthermore, SearchBar allows users to store notes in their query context, enabling them to persist information that could be useful for task resumption. In their study, Morris et al. (2008) found that only a few participants of the control group (i.e., the group without their tool) created bookmarks, with only one participant creating bookmarks relevant for task resumption. Overall, participants that had used their tool reported significantly less repeated work when trying to re-find relevant artifacts (Morris et al., 2008).

The approach developed by Leiva (2011), MouseHints, is an exploration of supporting users in regaining task context when switching back to a tab they had previously used but switched away from. MouseHints displays a visual representation of previous mouse interactions and the last location of the mouse before the tab was left so that users can understand how they had previously interacted with the tab. When evaluating their approach, Leiva (2011) found that study participants resumed tasks three times as quickly when provided with visual support (Leiva, 2011).

In contrast to previous approaches, Hahn et al. (2018) developed an approach to support the organization of complex search tasks in mobile browsing and did so utilizing a dedicated browsing environment (Bento Browser). Their approach seeks to address challenges that are prevalent in mobile environments, where screen size is severely constrained and work is performed in short bursts with regular interruptions. Bento is organized around the concept of tasks and subtasks instead of tabs, as Hahn et al. (2018) argue that the applicability of tabs for task-based work is severely limited, especially in mobile environments (Hahn et al., 2018).

Chapter 3

Approach

To evaluate our research questions, we designed and developed an artifact grouping approach that integrates with web browsers and allows users to curate and interact with tab groups. Our approach consists of two major parts: a tab grouping interface that provides visuals and interactions for tab group organization, and a heuristics engine that monitors browser activity and computes tab group suggestions. In the following, we provide a short overview of our grouping interface (Section 3.1) and elaborate on the approach to automated grouping that powers our heuristics engine (Section 3.2). Furthermore, we present how we incorporate user feedback to improve suggestions (Section 3.3). Finally, we also provide a summary of how we implemented our prototype (Section 3.4).

3.1 Tab Grouping Interface

Our tab grouping interface is the primary point of interaction for users. Its overall design was driven by the need of having an easily usable but tightly integrated method of grouping artifacts, such that users can easily incorporate the tool into their workflow. To achieve the necessary level of integration, the interface has been implemented as a browser extension for common browsers, preventing unnecessary context switching to other applications. Overall, the tab grouping interface has been designed to allow for more fluid interactions when compared to using built-in artifact management systems like bookmarks and history.

To support tab management and task switching in general, tabs can be added to tab groups and single tabs or entire tab groups can be opened and closed, with open tabs being highlighted. Artifacts can easily be (re-)organized by dragging them between any of the parts of the view (e.g., from the current tabs into a new or existing tab group). A focus mode can be activated to allow for a single concurrent tab group to be active, meaning that other active groups would be closed when a new one is opened. Furthermore, the tab grouping interface is available as a dedicated overview that can be used to view and organize all tab groups (see Fig. 3.1), and as an integrated sidebar that can be opened while working in another tab (see Fig. 3.2)¹.

In addition to supporting manual artifact grouping, the browser extension optionally integrates with our heuristics engine to support the user with automated group suggestions. Once the engine is enabled by the user, a background process starts observing browsing behavior and deriving suggestions. The upcoming Section 3.2 goes into more detail regarding the derivation of automated tab group suggestions.

¹An introduction to key extension features is available as feature documentation (see Appendix C).

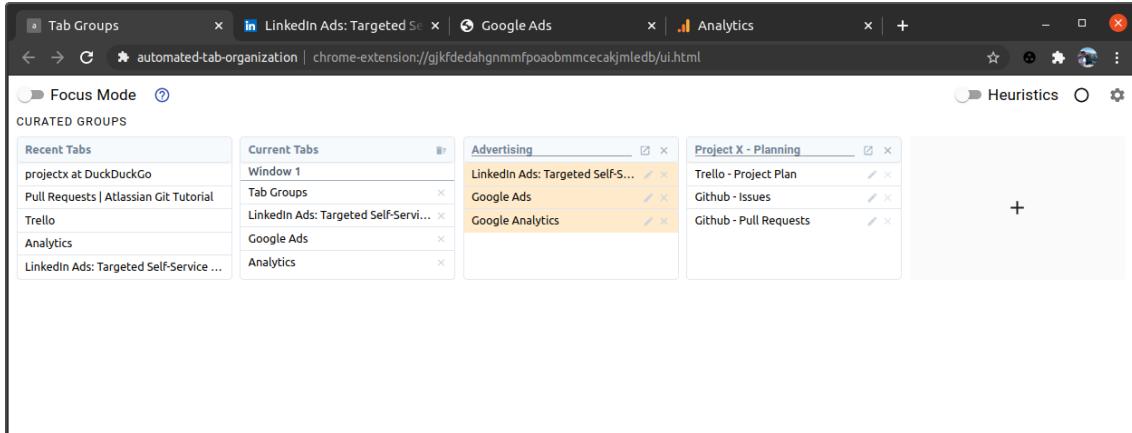


Figure 3.1: Tab Grouping Interface: Overview. The overview of our tab grouping interface displays the current tab state of the browser in terms of tabs that are currently open and tabs that have been closed recently. Furthermore, the interface allows for the creation of manual tab groups and enables subsequent interactions with these groups. When enabled, the interface further enables interactions with suggested tab groups. *Source: Own depiction.*

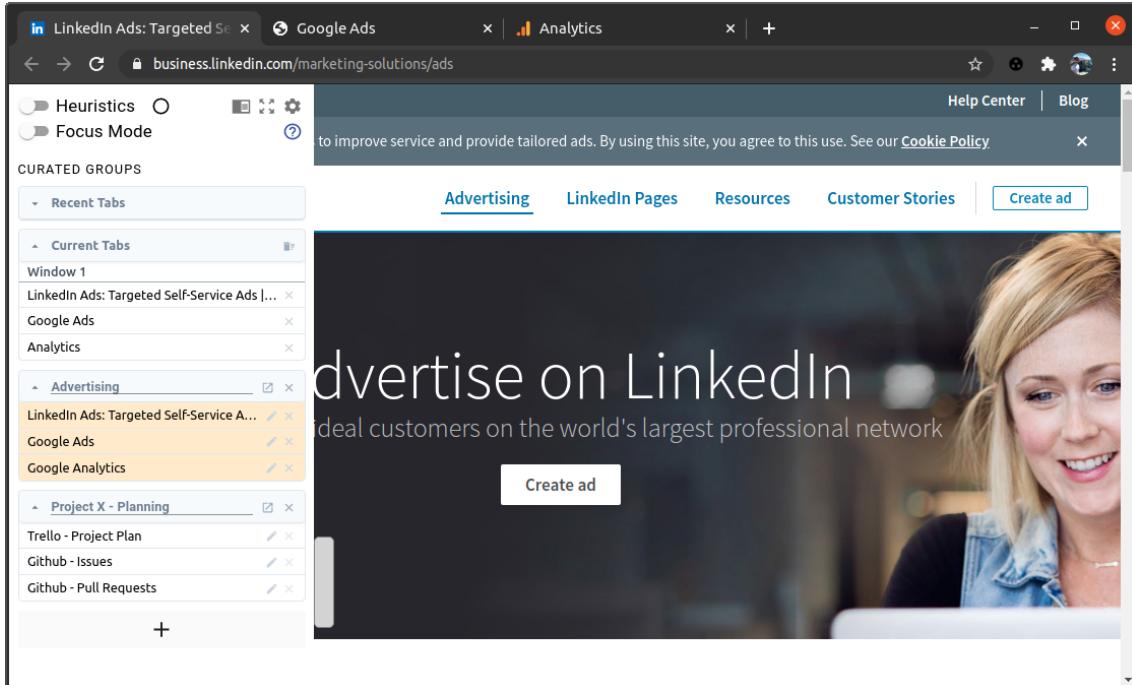


Figure 3.2: Tab Grouping Interface: Sidebar. The sidebar view of our interface offers the same functionality as the large overview but can be opened on any site that the user is currently working on. To toggle the sidebar, users can use a dedicated toggle button or use a customizable keyboard shortcut. *Source: Own depiction.*

3.2 Automated Tab Grouping

The second part of our approach is the *heuristics engine*, which monitors browser activities and computes tab groups that could be relevant to the user. In general, our approach is centered around the assumption that related tabs will often be used concurrently or in temporal sequence, and that there will be regular switches between them.

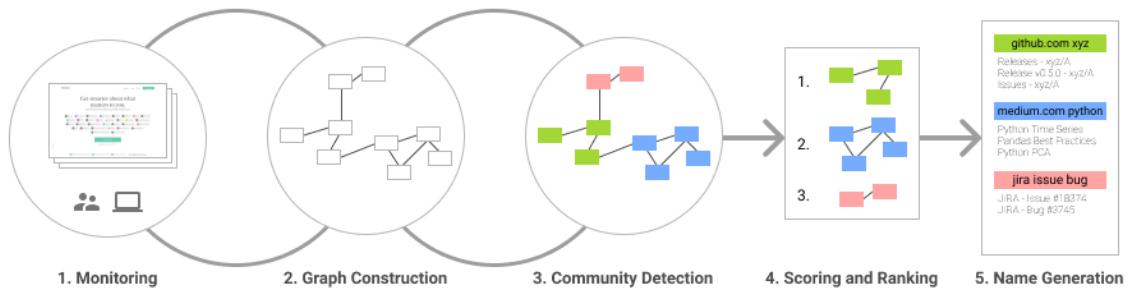


Figure 3.3: Overview of the Automated Tab Grouping Process. The overview visualizes the most important steps of our automated tab grouping approach. *Source: Own depiction.*

The approach implemented by the heuristics engine roughly follows the steps as depicted in Fig. 3.3:

1. Continuous Monitoring of Browser Work
2. Construction of a Tab Switch Graph
3. Community Detection
4. Scoring and Ranking of Communities
5. Creating Tab Groups with Automated Naming

We will dive deeper into each of these steps in the following sections and explain the process in terms of a simplified synthetic example. Furthermore, to improve the comprehensibility of our approach, the necessary background is shortly summarized where needed.

3.2.1 Monitoring

As our tab grouping approach is based on user behavior, all browser events are gathered by our browser extension and redirected to the heuristics engine for further processing. Examples for such events include the opening and closing of tabs, switches between open tabs, or updates on the contents (i.e., target website) of a tab. Relevant interactions with tab groups (e.g., opening or closing a tab group) or suggestions (e.g., accepting a suggestion) are handled similarly. Finally, the current internal state of the browser (e.g., currently open tabs) and our extension (e.g., the list of curated groups) is regularly synchronized with our heuristics engine to prevent inconsistencies and allow for the computation of additional statistics.

3.2.2 Graph Construction

The central data structure of our system is a graph of browser tabs (as vertices) and tab switches (as edges between vertices). Each time the user performs a tab switch in the browser (e.g., changing between two open tabs or opening a new tab), the vertices and a new edge are added to the underlying graph, or the weight of an existing edge incremented.

Our implementation is based on an undirected graph, as we assume that a relationship between two tabs is relevant independent of the direction of the switch. The strongest relationships between tabs will have the highest edge weight, as switches back-and-forth are aggregated onto the same edge. This allows for easier processing with algorithms that do not necessarily work on directed graphs. Additionally, the graph is enforced to be acyclic, as having switches between two instances of the same tab would not increase the information available for grouping.

Fig. 3.4 visualizes a mock example of a switch graph that could have been collected through browsing activities. We will work through the steps of our approach based on this example, though some steps might be slightly simplified for better comprehensibility.

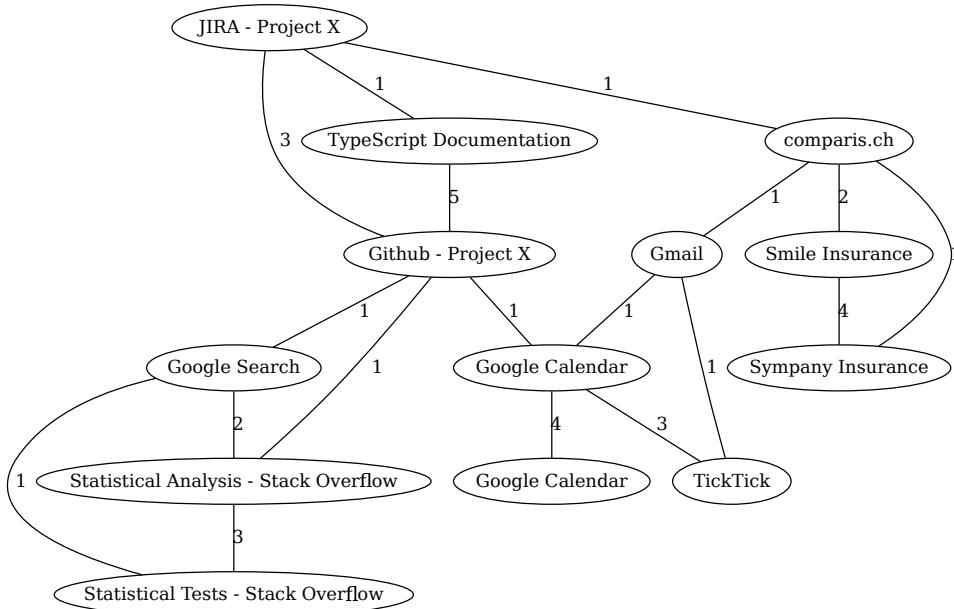


Figure 3.4: Graph of Tab Switches for our Synthetic Example. Each vertex (i.e., circle) is representative of a tab that has been visited at some point. Edges between vertices represent the switches between the respective tabs, irrespective of their direction. The occurrence count for each switch is marked along the edge. *Source: Own depiction.*

Reweighting

When preprocessing the tab switch graph, we apply an edge reweighting procedure to ensure that certain edges are given more significance when tab groups are detected, or conversely, to ensure that certain edges are filtered and removed before tab groups are detected. More specifically, we compute and apply both a *decay factor* to allow for expiration over time, and a *similarity factor* to increase the probability of grouping artifacts that are related on a higher level, not just in terms

of having very similar URLs. Pages with exceedingly similar URLs are often explicitly linked anyways, so grouping them provides less of a benefit than grouping high-level pages.

Decay Factor We compute a decay factor (F_{decay}) to reduce the weight of tab switches that have not occurred in a long time when compared to switches that have been used recently. In our default parametrization, the decay factor is set to zero for switches that have not occurred in two weeks, and scales linearly within $[0, 1]$ for switches that have been used more recently.

We can formulate the expirationFrontier (i.e., the timestamp at which older switches become irrelevant) as the difference between the current timestamp in seconds (nowTs) and a two-week duration in seconds (expirationWindow):

$$\text{expirationFrontier} = \text{nowTs} - \text{expirationWindow}$$

The decay factor incorporates the timestamp of the last occurrence of a tab switch (lastUse), as well as the expirationFrontier and current timestamp (nowTs):

$$F_{decay} = \begin{cases} 0 & \text{if } \text{lastUse} < \text{expirationFrontier}, \\ \frac{\text{lastUse} - \text{expirationFrontier}}{\text{nowTs} - \text{expirationFrontier}} & \text{otherwise} \end{cases}$$

Similarity Factor The similarity factor ($F_{similarity}$) is based on the notion that very similar tabs will be less useful when grouped, which was a feedback that we had received repeatedly in pilot testing (e.g., if only a single digit changes in the URL). In our default parametrization, tab switches on the same origin (e.g., `github.com`) that are exceedingly similar based on the Levenshtein² distance metric ($D_{Levenshtein}$, $[0, 1]$) will result in a similarity factor of zero. All other switches result in a factor of one minus 40% of $D_{Levenshtein}$.

$$F_{similarity} = \begin{cases} 0 & \text{if } \text{sameOrigin} = 1 \& D_{Levenshtein} > 0.95, \\ 1 - 0.4 * D_{Levenshtein} & \text{otherwise} \end{cases}$$

Weight Updates Based on the values of both F_{decay} and $F_{similarity}$, the new weight (W_{new}) of each edge is computed with a simple product of initial weight (W_{init}) and factors:

$$W_{new} = W_{init} * F_{decay} * F_{similarity}$$

When applying the reweighting, our exemplary graph would be updated as shown in Fig. 3.5. For the purposes of this example, we have set the similarity between tabs to 0.05 for unrelated tabs (to account for similarities in e.g. the top-level domain) and to estimates for related tabs (e.g., 0.96 for Google Calendar). Additionally, we set the decay factor for all tabs to 1, thereby removing the influence of time on the graph (to reduce complexity).

Filtering

As community detection algorithms try to group all vertices in a given graph, the graph must not contain large amounts of spurious vertices and edges. Therefore, we apply a graph filtering procedure that results in a reduced graph containing only significant vertices and edges. Overall, this reduces the size of the graph, the number of communities that are detected, and also the load of all subsequent processing. Furthermore, it ensures that the generated tab groups consist only of tabs that are above a certain level of relevance.

²Binary codes capable of correcting deletions, insertions, and reversals. Levenshtein (1966).

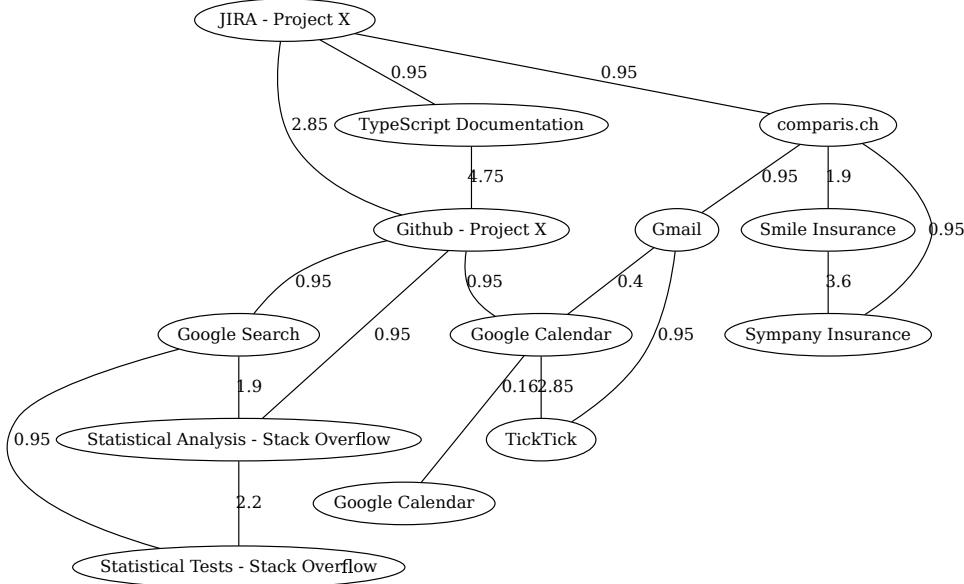


Figure 3.5: Reweighted Tab Switch Graph for our Synthetic Example. Each vertex (i.e., circle) is representative of a tab that has been visited at some point. Edges between vertices represent the switches between the respective tabs, irrespective of their direction. The updated weight for each switch is marked along the edge. *Source: Own depiction.*

Our preprocessing is based on the premise that we cannot distinguish between accidental switches³ and relevant switches when such a switch occurs only very few times. We enforce a minimal edge weight of at least two in our default parametrization, ensuring that spurious one-off switches will not be included in the graph, as well as removing any tab switches that are lowered in weight significantly by either of the decay of similarity factors. Additionally, if any of the tabs linked by a filtered spurious switch become completely disconnected, they will not be included in the final graph, thus resulting in a lower overall count of vertices.

When we apply the filtering procedure with default parameters (i.e., removing weights below two), we end up with a tab switch graph as shown in Fig. 3.6. In a real-world example, the resulting graph would not be all disconnected components, as not all connections between components would be filtered out after prolonged usage.

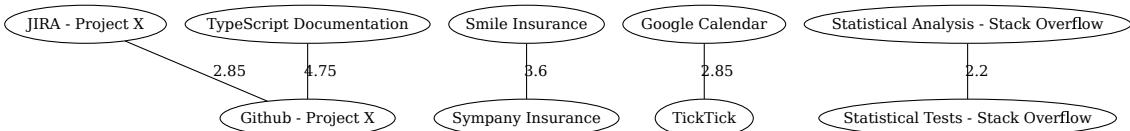


Figure 3.6: Filtered Tab Switch Graph for our Synthetic Example. Filtering has been applied in our default parametrization with a minimal edge weight of two. *Source: Own depiction.*

³We define accidental switches as cases where users switch to a tab mistakenly. For example, accidental switches can occur because users do not remember which tab contained relevant information and need to click through several tabs to retrieve the relevant one.

3.2.3 Community Detection

Community Detection in General Graphs are a data structure that is critical in many scientific disciplines, as they can represent complex processes and systems. When such graphs represent a real-world system, they often exhibit a *community structure*, that is, they consist of *communities* that are well-connected in themselves, but relatively separate from other communities. A property that is required so that a group of vertices can become a *community* is the *connectedness*, meaning that each vertex within a community must be reachable from the other vertices using edges that lie within the community (Fortunato, 2010).

The identification of communities has become an important computational problem, especially as the size of graphs to be processed has increased significantly. The main goal of *community detection* approaches is, therefore, to analyze graphs regarding their community structure as efficiently as possible. After applying a community detection approach on a graph, the result is most often a set of *partitions*, which are sets of vertices that cannot overlap (i.e., a vertex can only be part of one partition). However, in real-world applications, vertices can often be associated with multiple groups, so some methods compute *covers* (i.e., sets of vertices that are allowed to overlap) instead (Fortunato, 2010).

The analysis of graphs regarding communities is often a problem with high computational complexity, which is why most algorithms compute an iteratively refined approximation of the community structure rather than attempting to find an exact solution (Fortunato, 2010).

Flow-Based Community Detection One of the many categories of community detection approaches are *flow-based* methods, where *flow* refers to the dynamics (e.g., a flow of information) that take place between components of a real system, and, therefore, along edges of that system's graph. Flow-based methods analyze the graph to find structures (i.e., communities) that capture flow, meaning that once such a structure has been entered, flow remains within the structure for a while (Bohlin et al., 2014).

One example of a flow-based method is the MapEquation framework as introduced by Rosvall et al. (2009). Their approach is based on the idea that a random walker traversing the graph stays within partitions much more often than it switches between partitions (i.e., as is the core idea of *flow*). When the path of the random walker is traced, a stay within a partition could be compressed instead of tracing every step within the partition, whereas a switch between partitions needs to be explicitly logged. The overall algorithm optimizes towards the best partitioning based on the information needed to compress the entire infinite traversal of a random walker. More intuitively, the better the partitioning of the graph, the fewer switches a random walker will perform over the course of its infinite traversal, and the less information required to describe its (infinite) path (Rosvall et al., 2009; Bohlin et al., 2014).

Map Equation for Signed Networks (SiMap) In our approach, we leverage a method developed by Esmailian and Jalili (2015) that is based majorly on the MapEquation framework (Rosvall et al., 2009). Esmailian and Jalili (2015) extend the MapEquation framework with capabilities for dealing with negative edge weights, which gives us a lot of flexibility in terms of parametrization. While our final approach did not include such negatively weighted edges, future extensions could incorporate negative user feedback as such, thereby providing strong constraints to the algorithm. Also, Esmailian and Jalili (2015) provide a reference implementation of their SiMap approach that we could reuse in our work⁴.

Community Detection on the Tab Switch Graph Once the tab switch graph is ready for processing, we apply the SiMap algorithm as previously described. From the communities that are derived

⁴The library code is available at <https://github.com/pouyaesm/signed-community-detection>.

by the algorithm, we filter the smallest as well as the largest communities, as these tend to not be useful for usage as a tab group. Larger communities tend to group many tasks, while communities with only one or two artifacts would often need to be merged to reuse. Therefore, communities are only processed further if they contain between 3 and 10 vertices.

When we apply community detection to our example graph, the outcome is trivial, as all components are disconnected from one another. However, in a real example graph that is much larger and more thoroughly connected even after filtering, the community detection algorithm would have to do work to come up with a valid partitioning. We would end up with the following groups in our trivial example:

Group 1: JIRA - Project X, TypeScript Documentation, Github - Project X

Group 2: Smile Insurance, Sympany Insurance

Group 3: Google Calendar, TickTick

Group 4: Statistical Analysis - Stack Overflow, Statistical Tests - Stack Overflow

We will continue using this example for the remainder of the process, as the final steps are non-trivial even on this example, serving the purpose of explanation very well.

3.2.4 Scoring and Ranking

As a further step in improving the suggestions provided to users, we rank the communities computed by the algorithm with a custom scoring function. We use the score as a derived measure of the confidence we have in a group, as well as a measure for the relevance of a group overall, allowing us to show the user the groups that we deem the most relevant and accurate.

The score of a group is derived from three equally weighted normalized components: the connectedness of the group ($C_{undirected}$), the average weight of edges between members of the group ($W_{avg,N}$), and the average PageRank⁵ of members in the group ($PR_{avg,N}$)⁶. While we include the first two measures to derive our confidence in the accuracy, the PageRank is included to improve the score of groups that include relevant tabs. Due to each item being normalized to $[0, 1]$, the score is bound to a range of $[0, 3]$.

More specifically, the connectedness, or *intra-cluster-density*, $C_{undirected}$ of an undirected (sub-)graph is defined as the ratio of potential edges within a graph that are actually present (Fortunato, 2010):

$$C_{undirected} = \frac{N \text{ edges}}{N \text{ potential edges}} = \frac{N \text{ edges}}{N(N-1)/2}$$

The formula for our custom score S is defined as follows:

$$S = PR_{avg,N} + W_{avg,N} + C_{undirected,N}$$

When applying the described scoring approach to our example tab groups, we obtain the following scores and ranking:

1. **Group 2** with $S = 0.46 + 0.88 + 1 = 2.34$
2. **Group 3** with $S = 0.46 + 0.41 + 1 = 1.87$
3. **Groups 1 and 4** with $S = 0.46 + 1 + 0 = 0.46 + 0 + 1 = 1.46$

⁵The PageRank Citation Ranking: Bringing Order to the Web. Page et al. (1999).

⁶The PageRank of each vertex is precomputed on the entire graph, ensuring that its relevance depends on all other vertices, not only the ones in the same graph.

3.2.5 Name Generation

To reduce the cognitive load for a user interacting with our suggestions, we derive an automatically generated summary for each community and provide this summary as a customizable title in our user interface. The automated title is computed using simple keyword extraction from the titles and URLs of all tabs in a group. To further improve the titles, we apply tokenization and pruning of stop words. The summary title is regarded as an approximation and thought to be replaced with a custom title when a suggested tab group is accepted by users. Therefore, we chose to go with a basic but functional approach.

Applying the procedure to our simple example, we would end up with the following titles:

Group 1: project x jira github⁷

Group 2: insurance smile sympany

Group 3: google calendar ticktick

Group 4: statistical stack overflow analysis

After name generation, tab groups are ready to be suggested in the browser extension. An exemplary tab group suggestion as shown in the user frontend could be visualized as shown in Fig. 3.7.



Figure 3.7: Exemplary Tab Group Suggestion. A suggested group as shown in the tab grouping interface. Users could interact with such a suggestion by saving it to their list of curated groups, by discarding the group or single tabs thereof, or by opening the group or any of its tabs. *Source: Own depiction.*

3.3 User Feedback

To improve the performance of our heuristics, it is crucial that user feedback is appropriately incorporated in the underlying data structure, thereby ensuring that badly rated suggestions will not be shown again. When we show a list of suggestions to the user, we allow for accepting (i.e., persisting) a group in one's curated groups, as well as for discarding a group, providing a rating and reason for the discard action (as shown in Fig. 3.8).

Based on the reason for the discard, we either mark the edge in our graph as discarded, ensuring it will never be incorporated in the graph again, or we reset its weight, ensuring it will be filtered in preprocessing until it is used again. In addition to accepting or discarding a suggested group in

⁷"project x typescript documentation" would be equally probable, as keywords with the same occurrence count are selected at random.

How appropriate was this suggestion?



Why are you discarding it?

WRONGLY GROUPED
NOT USEFUL AT THIS TIME
OTHER REASON

DO IT!

Figure 3.8: Feedback Dialog shown on Suggestion Discards. For each group discarded by users, we asked them to rate the suggestion on a scale of 1 to 5, as well as to provide a reason for the discard.
Source: Own depiction.

completion, users may also accept or discard single tabs. However, these interactions do not yet influence our underlying data structure.

3.4 Prototype

To support the evaluation of our approach, we developed a prototype that users could install on their own device. Our prototype implementation consists of two key parts: a browser extension providing the tab grouping interface (Section 3.1), and an application that runs locally in the background to execute the heuristics engine (Section 3.2).⁸

Browser Extension We built our browser extension on standards that allow an installation in most common browsers (i.e., Chrome, Firefox, and Microsoft Edge)⁹ so that we could support the largest possible user base. The extension was designed to be the central entry point for any activities that users needed to perform regarding our approach and study. Furthermore, we incorporated a step-for-step tutorial that guided users through the setup process for the heuristics engine, and allowed users to control the engine from the extension as well (e.g., to pause and resume tracking). Additionally, we included reference documentation of the most important features and functionalities that was reachable from any of the extension views.¹⁰

Heuristics Engine Our heuristics engine is a native application that runs locally and integrates with the browser extension to track browsing activities and provide group suggestions. After an initial setup with the tutorial as described above, the heuristics engine would start or stop processing when a browser session is opened or closed, respectively. For privacy reasons, all data collected by the heuristics engine is only stored on the local machine, and in a location that is communicated to the user. As our application is based on the Java Virtual Machine (JVM) and cross-platform compatible, we were able to provide packages for all major operating systems (i.e., Windows, Mac OS, and Linux)¹¹.

⁸Appendix A provides more detailed information on the technologies used in our prototype.

⁹Safari requires a custom implementation and is not supported at this time.

¹⁰The full contents of the tutorial and feature documentation are included in Appendices B and C.

¹¹While the prototype works on Windows, we found some issues that could degrade device performance: the background process is currently not always closed when the browser session ends, especially when using Chrome.

Chapter 4

Study Method

To evaluate the applicability of our tab grouping approach, as well as tab grouping in general, we performed a field study covering two workweeks. We report on the setup and procedures of the study in this chapter.

4.1 Procedures

We performed an *in situ* two-week user study including statistical data collection, an intermediate interview after the first week, and a concluding interview and survey.¹ Our primary user-study was organized in two phases: baseline phase and intervention phase, and was preceded by a short pilot study.

Pilot Study We conducted a small-scale pilot study with two participants to evaluate the usability of the system in terms of installation and documentation, as well as to evaluate the performance of the heuristics when exposed to a different set of workflows (compared to self-experimentation among the researchers). Additionally, we conducted short interviews to evaluate the comprehensibility and validity of our semi-structured interview. The feedback gained regarding usability and performance, as well as the responses in the interviews, were used to further iterate on these parts of the study. Furthermore, as both participants had no relation to software engineering, we could gather very useful feedback especially regarding setup, usability, and general preparations.

Baseline Phase In the week-long baseline phase, participants installed our browser extension and heuristics engine at the start of the week, after which they continued working as normal for the remainder of the week. The statistics collected during this phase provide a baseline (i.e., benchmark) for statistical comparisons, as participants were not influenced by us or our extension in any way (all functionalities for grouping were invisible and disabled). Additionally, the baseline phase also served as a warm-up period for the grouping heuristics, as the browser behavior was already being tracked and groups were being computed. We could thus directly show participants a list of suggestions at the beginning of phase two.

Intermediate Interview The baseline phase was concluded with a semi-structured intermediate interview, in which we asked participants about their browsing behavior and tasks they normally worked on in a browser. Additionally, we walked participants through the activation of grouping functionalities and explained the most important concepts. Furthermore, we asked participants

¹The welcome document in Appendix F shows the steps that participants needed to perform to get started.

to give us their first thoughts on the initial suggestions that had been accumulated throughout the baseline phase. Due to the timing of the interviews, some participants had their intermediate interview (and activation of grouping functionalities) on the last day of the first week, while others had it on the first day of week two.

Intervention Phase The second phase of the study, the intervention phase, was another week that participants spent working normally while additionally trying to incorporate our grouping functionalities into their everyday workflows.

Follow-Up Interview We concluded the intervention phase with a semi-structured follow-up interview, in which we asked participants about their experience with our functionalities and potential usability issues that could hinder/invalidate our analysis. Additionally, we asked participants to name us a few examples of groups that stood out, either positively or negatively, as well as the reasons for this perception. Finally, we instructed participants on the data submission procedures and what data we would need to perform our analysis, thus preparing them to review and submit the data on their own.

Follow-Up Survey After the conclusion of the active phase of the user study, we sent out a final follow-up survey that was focused on the usability of the functionalities as well as more fine-grained aspects regarding the performance of suggestions and other features. We evaluated the usability of our systems using the System Usability Scale (SUS) (Brooke, 1996) and asked participants about their expectations of a tab grouping tool if they were to use it in the future.²

4.2 Participants

The participants of our pilot (2 participants, female) and primary study (6 participants, male) were recruited from the personal and professional contacts of the researchers. All participants remained engaged over the course of the entire study, including all of its phases and activities, and submitted their data for further analysis. The participants of our primary study were all related to computer science in some form (e.g., computer science students or software engineers), while the participants of the pilot study were knowledge workers working in Finance.

The participants of our primary study were aged from 26 to 29, with a mean of 27 years and a standard deviation of 1.27 years. The work experience of participants ranged between three and six years, with a mean of 4.66 years and a standard deviation of 1.21 years. For their profession, five participants mentioned working as a software engineer, three of which also mentioned studying computer science. One participant mentioned working as a research assistant. When additionally surveyed about their role, five participants mentioned working as a software engineer, while one participant described their role as full-stack project management. The participants of our pilot study did not partake in our follow-up survey.

4.3 Data Collection

The data collected and derived from participants' browser and tab grouping usage includes statistical aggregates on various browser-related characteristics and suggestion interactions, as well as snapshots of suggested groups over the course of the intervention phase. More specifically, we have collected statistical aggregates in 30-second increments for the following features:

²The full contents of our follow-up survey are available in Appendix E.

- The number of open tabs and windows
- The average and standard deviation of tabs per window
- How many of the open tabs are or are not part of a group (manual or suggested)
- How many tab switches were performed (within groups, between groups, from a group, to a group, or outside of groups)
- The numbers of curated and suggested groups
- How many curated groups are opened or closed
- How often focus mode was used (i.e., how many groups were opened during focus mode)
- How many suggested groups or tabs were accepted or discarded
- The age of tabs (i.e., time since creation) in minutes
 - Binned, one of [<1, 1-2, 2-5, 5-10, 10-30, 30-60, >60]
- The staleness of tabs (i.e., time since last use) in minutes
 - Binned, one of [<1, 1-2, 2-5, 5-10, 10-30, 30-60, >60]
- Timings of performed tab switches in seconds
 - Binned, one of [<1, 1-2, 2-5, 5-10, 10-30, 30-60, >60]
- The numbers of discarded groups given a specific rating
 - Binned, one of [1, 2, 3, 4, 5]
- The number of groups discarded with specific reasons
 - One of [WRONG, NOT_USEFUL, OTHER]

In addition to the aggregates as listed above, and if participants agreed to share this data, we also collected:

- A (censored) version of the tab switch graph
- Tab events and tab switches (censored)
- Snapshots of suggested groups in 1-hour increments (censored)

4.4 Data Analysis

Open Coding Over the course of our user and pilot studies, we performed 12 interviews with primary participants (one intermediate and one follow-up interview each), and 2 interviews with pilot participants with all questions combined. These interviews were held in Swiss German and subsequently transcribed to German for further analysis. Based on our transcripts, we applied an open coding approach (Miles and Huberman, 1994) in which we categorized all statements made in our interviews, and translated all statements to English to allow for inclusion in our study. Overall, we coded 447 statements in 7 high-level domains: Work Fragmentation, Browser Work, Tab Grouping, Workflow Adaption, Group Suggestions, Overall Experience, and Future Work. Domains contained between 14 and 181 elements and were subdivided into 33 lower-level topics. Where appropriate, these topics were further divided into small subtopics with only a few elements.

Statistical Analysis The statistical aggregates we collected throughout our user study were primarily used to evaluate whether the phases of the study exhibited interesting patterns or trends in their development. For example, we evaluated the development of open tabs across both phases of the study to see whether the number of concurrent tabs had reduced in the intervention phase. Where appropriate, we performed statistical tests with a significance level of 5% ($\alpha = 0.05$).

Chapter 5

Results

During our field study, we interviewed and surveyed participants about their work and browsing behavior, the problems they encounter when working in a browser, and how they were able to incorporate tab grouping in their workflow. We developed hypotheses for patterns that might emerge when participants successfully apply tab grouping and statistically analyzed the data to evaluate these hypotheses. Our results are structured in terms of our research questions as follows:

RQ1: Can grouping browser tabs support knowledge work?

RQ2: Can we automatically group browser tabs to support knowledge work?

a: Can we accurately group browser tabs that are related based on knowledge workers' workflows?

b: Can we use our identified tab groups to support the knowledge work in the web browser?

Whenever we refer to specific participants, we use *Pi* to signify a pilot participant, while *Pa* serves the purpose of identifying a primary study participant. For example, *Pa4* would be the participant of the primary study with the identifier 4. Please note that the numbers of participants are randomized in that their identifiers do not match between survey, interviews, and other collected data.

5.1 RQ1: Can grouping browser tabs support knowledge work?

As part of our first research question, we surveyed participants on their work behavior and fragmentation (i.e., interruptions and task switches), as well as on their work within browsers and what challenges they face while performing browser work. To evaluate whether tab groups could be a helpful abstraction in supporting knowledge work, we further evaluated the types of tasks that participants perform in the browser and whether they could imagine grouping their tabs to support them with their tasks. Finally, we analyzed whether participants had interacted with our system to incorporate tab groups in their workflow.

5.1.1 General Knowledge Work

The majority of participants work on multiple tasks in parallel, and regularly switch their active task. While the majority of participants often work on multiple tasks in parallel, they also deliberately try keeping the number of parallel tasks low: "I try to keep it low, I would say I have

two to three simultaneous tasks on a typical morning” (Pa3). One participant described focusing on one project for the workday: “I try to focus on one project per day, maybe a second one in the evening” (Pa6), and another mentioned that “one project is often very much in focus, while the others are just worked on besides” (Pa1). While some participants described switching tasks “frequently” (Pi2) or “about every thirty minutes” (Pa3), other participants stated that “there are times where they are very focused and work on one single task” (Pa4). Overall, as shown in Fig. 5.1, four participants agreed or strongly agreed that they “work on a number of projects/tasks simultaneously”.

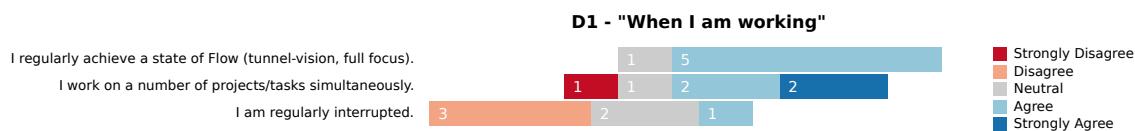


Figure 5.1: Follow-Up Survey Question D1: “When I am working”. Source: Own depiction based on responses as given by our six primary participants.

Participants generally manage to deal with distractions and achieving a state of Flow, but their specific strategies vary. While deliberate focus on one project or task might be a good goal, it can be hard to achieve because of interruptions, distractions, and externally triggered task switches (e.g., urgent issues). Some participants described “not having many distractions as they work on their own” (Pa4) but still “sometimes distracting themselves” (Pa4). One participant mentioned deliberately “having turned off notifications of all services” (Pa2) and “trying to only look at messaging once an hour” (Pa2). Other participants work with activated notifications as they “don’t like missing messages in work chat for an hour” (Pa1) or can “just ignore notifications when working” (Pa5). Only one participant mentioned using an external system for structured focus: “I currently try to focus for 30 minutes at a time by setting an alarm clock, after which I am allowed to do some distracting things” (Pa6), as well as that they “purposefully don’t try to focus for hours at a time, as that leaves them drained for the rest of the day” (Pa6). As can be seen on Fig. 5.1, five participants agreed to the statement that they “regularly achieve a state of Flow”. Only one participant agreed that they were “regularly interrupted”.

5.1.2 Browser-Based Knowledge Work

Participants work with their browser for a significant portion of their workday. When asked about the portion of their work time that is spent in their browser (excluding web development on localhost), participants mentioned different numbers or ranges varying between 30% (Pa2) and 80% (Pi2) with means from 49.4% (lower range borders) to 53.8% (higher range borders). Some participants stated that the browser is such a critical tool that they “almost always have a browser open, as they need a lot of tools that are only accessible using the browser” (Pa6).

Some participants rely on elaborate bookmarking approaches for long-term persistence and task resumption, while others rarely use bookmarks and memorize most of their artifacts. The majority of participants states using a “bookmark bar with folders” (Pa1, Pa2, Pa4, Pa5) to organize their artifacts, with folders being “more or less sorted by project or context” (Pa4) in most cases. One participant stated that “with well-organized bookmarks, task resumption is easy” (Pi2) as an explicit advantage of such approaches. Conversely, other participants mentioned having a

“big chaos in their bookmarks” (Pa4) and “completely losing the overview in their nested folder structure” (Pa2). One participant mentioned “not really being a bookmark user” (Pa3), while another said that “I have only few bookmarks in my bookmark bar, as I know most pages by heart and don’t need a bookmark” (Pa6). One participant stated: “I use the tab grouping principle a lot, as can be seen in my bookmark bar” (Pa1), which goes to show that the tab grouping principle can be applied with built-in browser tools (e.g., bookmarks) and does not require elaborate tools. When asked about their usage of the browser history as the second artifact management system besides bookmarks, some participants mentioned “using the history rarely to look up something that they had previously used” (Pa1, Pa3, Pa6). One participant mentioned “using the history extensively and always trying to enter a relevant string to get to the needed artifacts” (Pa2) and that: “If my browser history were deleted, I would have a problem, as I would not find anything anymore” (Pa2). As an explicit hindrance towards using the history more often, one participant mentioned often not finding what was looked for as the history “is too full and verbose” (Pa1).

Developers perform many one-time tasks (e.g., search for specific problems) and often work with issue trackers, code repositories, and documentation resources. To evaluate whether participants regularly come back to the same artifacts (and could, thus, potentially group these artifacts), we surveyed our participants on the portion of their time spent on regular/repeated tasks (e.g., scheduling) compared to one-off tasks (e.g., looking for a specific solution on StackOverflow). Participants gave responses ranging from 25% to 50% with a mean of 37.5% and a standard deviation of 10.8%. Additionally, we asked our participants about the tasks they perform in the browser, extracted keywords, and ranked the keywords by their occurrence count. The keywords that were mentioned the most often by at least two participants are shown in Table 5.1, in two variants with (left) and without (right) pilot participants being included.

Keyword	Total	Unique	Keyword	Total	Unique
search	10	6	search	9	5
issue-tracking	7	3	issue-tracking	7	3
documentation	6	4	documentation	6	4
repositories	6	4	repositories	6	4
email	5	4	stackoverflow	4	3
scheduling	5	3	scheduling	4	2
stackoverflow	4	3	development	3	3
development	3	3	email	3	2
communication	2	2	communication	2	2
crm	2	2	localhost	2	2
localhost	2	2			
shopping	2	2			
task-management	2	2			

Table 5.1: Occurrence Count of Task Keywords: Total Count and Unique Count. Total count represents how often a keyword was mentioned overall, while unique count shows by how many participants a keyword was mentioned. Source: Keyword extraction from our interviews including pilot participants (left) and excluding pilot participants (right).

When we exclude responses from our pilot interviews, we get the set of keywords most frequently mentioned by our primary participants, which were all related to software engineering (as shown in Table 5.1). While the top tasks remain the same across both sets of keywords, excluding

the two pilot participants (both non-developers), caused the keyword for *email* to rank much lower in the list, and the keywords for *crm* and *shopping* to disappear.

5.1.3 Challenges in Browser-Based Knowledge Work

The vast majority of participants has issues in keeping track of their growing list of browser tabs. Most participants stated that their browser tends to grow more and more cluttered over time. While some attribute the problem to their being “bad at closing tabs” (Pa2) or “closing conservatively” (Pa6), others explicitly state that they “keep tabs open for reference” (Pi2), “as a reminder for an open task” (Pi1), or that they “don’t close previous tabs when switching tasks” (Pi1). Only one participant explicitly stated generally closing tabs after use: “I am not a person that has many open tabs. I mostly close the tabs after use” (Pa5). Most participants qualitatively stated having many tabs open concurrently, though the overall responses in our survey varied from 5 to 50 concurrent tabs with a mean of 19.5 tabs and a standard deviation of 15 tabs. Furthermore, when surveyed on the number of windows that they have open concurrently, we received responses ranging from 1 to 4 windows, with a mean of 2.25 windows and a standard deviation of 1.08 windows. As visualized on Fig. 5.2, the number of open tabs and windows varies a lot across participants but tends to stay within relatively narrow ranges for each individual. Only two participants display large variations in terms of open tabs and windows.

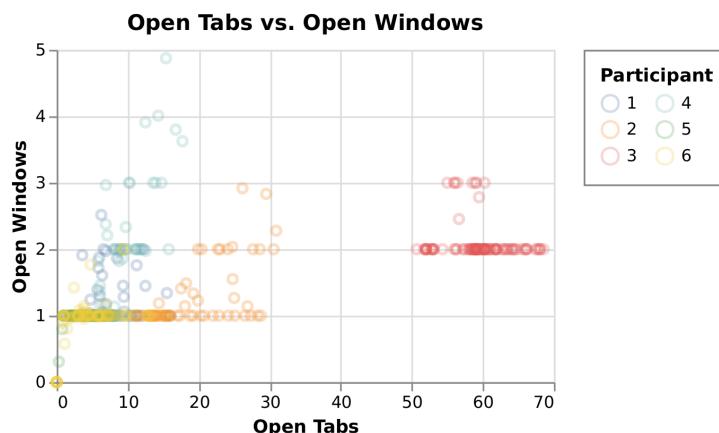


Figure 5.2: Number of Open Tabs plotted against Number of Open Windows for each Participant.

Source: Own depiction based on the interaction logs of our six primary participants.

Participants identified their main challenge in the cognitive load of keeping the overview of a cluttered browser environment. Participants mentioned that they “don’t like coming back to the PC with 1000 open tabs that they don’t need anymore” (Pa1) or that “once you have 20 or more tabs open, you constantly need to think about which tabs belong together” (Pa2). Furthermore, they mentioned different strategies of coping with said clutter, but most often end up closing everything periodically (Pa1) or when triggered by performance degradation (Pi1) or lost overview (Pa4, Pi1). An additional challenge that was mentioned by several participants concerns the issue of losing track of an important tab and having to search for it: “When I lose a tab, I look for it using a *Trial and Error* approach.” (Pa3).

5.1.4 Tab Grouping in Principle

Participants believe that tab grouping can improve productivity and support work organization. We asked participants about their opinion regarding tab grouping: could they imagine tab groups in general as a useful supporting mechanism for their actual work? All our participants stated, in one way or another, that they believe the tab grouping approach to be a useful concept. One participant described that “groups are sensible, even if just for the visual support. Using groups, what belongs together is clear much quicker” (Pa2), while another found that “groups are helpful for work organization” (Pi1). One participant described his applying the principle with bookmark folders: “I use the tab grouping principle a lot, as can be seen in my bookmark bar” (Pa1). When surveyed on their thoughts about tab grouping (as shown in Fig. 5.3), all participants agreed or strongly agreed that tab grouping “can improve productivity significantly”.

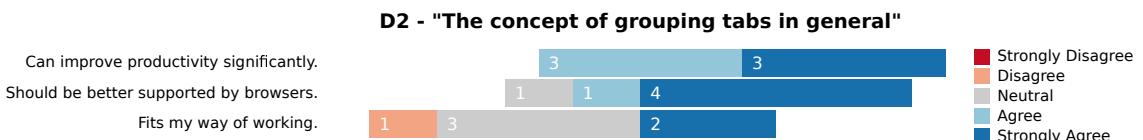


Figure 5.3: Follow-Up Survey Question D2: “The concept of grouping tabs in general”. Source: Own depiction based on responses as given by our six primary participants.

The usefulness of tab grouping depends on the individual workflow and tasks. While participants agreed on the potential use of the tab grouping concept, some participants voiced concerns that “tab grouping is a good idea in principle but it requires tasks that are generally repeated” (Pa3) and “I like the principle but found that it does not fit my workflow” (Pa3). These concerns were voiced in the context of a typical developer workflow that can include many search and one-time tasks (e.g., troubleshooting with StackOverflow): “I don’t have tasks A and B, that changes continuously. The developer workflow of doing something else every day” (Pa3). A related concern was voiced in that workflows of users vary a lot in general: “It would probably be very hard to get such a tool to fit everyone’s workflow” (Pa6). As can be seen on Fig. 5.3, only two participants strongly agree that tab grouping “fits their way of working”, while three did not give an explicit statement, and one participant disagreed. Five participants responded that native browser support for the concept should be improved and one participant mentioned liking the idea but that “tab grouping should be much more incorporated into the browser and bookmarks” (Pa5).

Developers generally imagine tab groups on a project- or issue-level (e.g., grouping application, documentation, and technical references). To figure out whether participants can profit from tab grouping, we asked them to provide us with good examples of tab groups that they could imagine for their work. We sought to find out whether the mental models of participants match in what could be grouped, as well as whether they would be able to represent these mental models when using our system. As we talked to developers in our primary study, responses were relatively tailored to development work. In general, groups were imagined on a project- or application-level so that they could be revisited over the course of developing: “A good group would be for things I revisit all the time, for example for an app I could have tabs for documentation, related examples, the repository.” (Pa4). One example of a lower-level grouping was tied to a specific implementation ticket: “The perfect example for a work group is a ticket, documentation, code, and maybe also a development or staging instance.” (Pa2). Another type of group named were groups for references:

"As a developer, I think it would be useful to have a group for each technology or framework" (Pa4). In piloting with other knowledge workers, we also got responses related to more generic contexts like "Marketing" (Pi1), "News" (Pi1), or "Shopping" (Pi1). Furthermore, one pilot participant mentioned wanting a "group with all services of a specific provider" (Pi2).

5.1.5 Adaptation of Tab Grouping

To find out in what use cases our system could be helpful, we asked participants about their usage of the manual grouping and interaction features. Additionally, we evaluated our quantitative data in terms of group interactions to see whether and how participants had created groups and subsequently interacted with them.

When introducing our system, we had talked about tab grouping in general and explained the most important features of the browser extension. However, we did not explicitly tell participants how they could, or should, incorporate the system into their workflow (e.g., no statements like "map your tasks to groups and open/close them to switch tasks"), as we wanted to get an unbiased view on how the system could or could not be helpful. That this lead to quotes like "I was not sure whether I had used the tool as was expected of me" (Pa4) seems to show that participants had themselves expected more direction. We did, however, set up a daily reminder that ensured participants saw the extension and suggestions at least once a day so that they would not completely forget about its presence.

The groups created by participants largely matched their initial ideas on useful tab groups. Some participants created groups for specific technologies in that they "created a group for the JAM-Stack" (Pa4) or grouped "resources for Tailwind CSS" (Pa2), while others created groups for specific implementation issues: "I have only created manual groups for the work context and in terms of groups for tickets with a corresponding name, where I could see all things related to a given ticket" (Pa2). One participant that had mentioned already applying tab grouping "represented their base bookmarks as a tab group" (Pa1). While our initial assumption was that participants would be primarily creating groups for long-term usage (e.g., tasks they work on near-daily over extended periods), one participant found the most use from "temporary groups" for cases where they "had found something at work that they were interested in personally, but did not want to look at at work" (Pa2) or where they could "store things when they had to leave, so that they did not have to leave everything open" (Pa2).

The number of tab groups created or saved by participants, as well as the portion of their work that was mapped to these groups, varied significantly. As shown by Fig. 5.4, the number of curated groups for each participant remained at zero during their respective baseline phase. After the activation of grouping functionalities, the number of groups increased almost monotonically and plateaued at a certain level for the majority of participants. One participant showed a steep upwards trend without a clear plateauing effect but the missing data does not allow us to make a conclusive statement towards further development. Only in one case did the number of groups go down after the plateau, indicating that the participant had removed groups that were previously present. We expected that the number of curated groups could plateau at a certain point where participants may have represented all of their current tasks, where an additional group might increase their cognitive load when looking at the grouping overview, or where they simply might lose interest in working with the extension.

We expected that the number of tabs that are grouped should increase with an increasing adaptation of our tab grouping approach, as more and more regularly used tabs are placed in groups either manually or via automated suggestions. When analyzing Fig. 5.5, we observe a significant difference between the time series of participants 2-4 and participants 1, 5, and 6, respectively. The

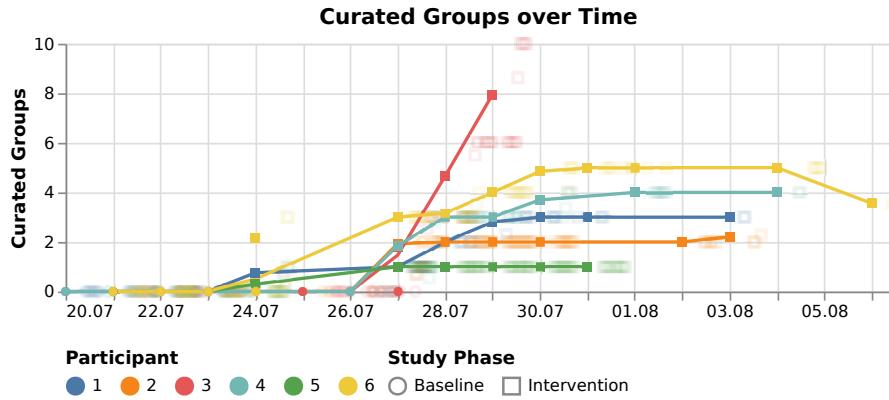


Figure 5.4: Development of Curated Groups over Time. The number of curated tab groups comprises both manually created groups and suggested groups that have been accepted. *Source: Own depiction based on the interaction logs of our six primary participants.*

first set of participants displays a clear increase in open tabs that were grouped, though the extent is different for each of the three participants. The latter group displays a very flat zero-oriented curve, meaning that very few or none of their open tabs were part of a group on average. A striking fact is that P6, who had the second-highest number of groups as per Fig. 5.4, simultaneously had very few open tabs that were in these groups. Even though these participants had few tabs open in general, the average being close to zero for the majority of the time reflects that their groups did not match what they worked on. When plotting the number of open tabs against the percentage of grouped open tabs (as visualized in Fig. 5.6), we can see that two participants remained largely within a range of 0-20% of grouped open tabs. Two other participants had much more significant variations with one participant covering the entire spectrum from 0-100% of grouped open tabs.

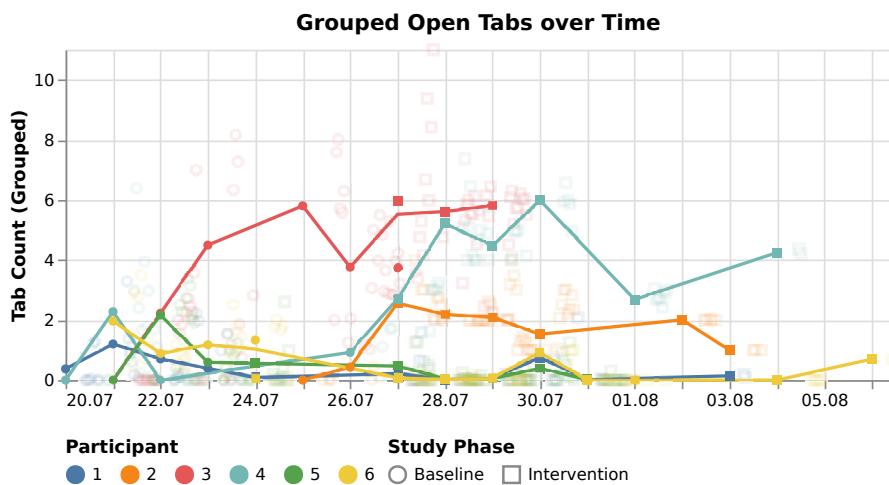


Figure 5.5: Development of Grouped Open Tabs over Time. *Source: Own depiction based on the interaction logs of our six primary participants.*

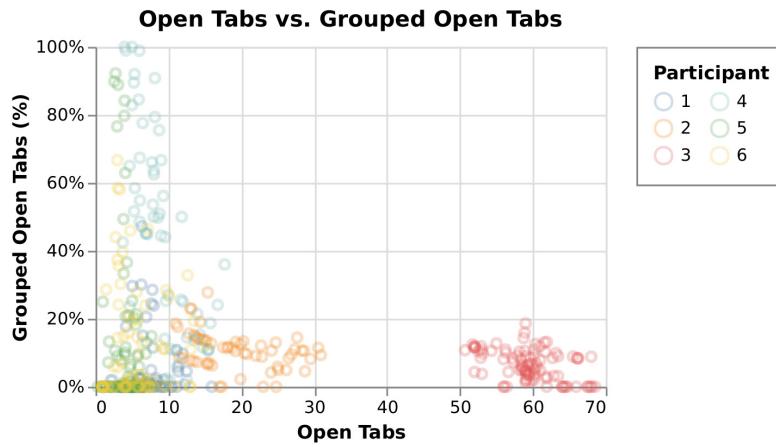


Figure 5.6: Open Tabs and Percentage of Grouped Open Tabs for each Participant. Source: Own depiction based on the interaction logs of our six primary participants.

Most participants interacted with groups primarily by closing them for cleanup, and only secondarily by opening them for task resumption. While we had built the system intending to support task switching and resumption, we had also hoped to achieve a less cluttered environment in the process. More specifically, we allowed participants to open entire groups to resume a task, to close entire groups of tabs once they were done with that task/group, and implemented a “focus mode” that would allow participants to switch between tasks/groups, automatically closing the previous and opening the new group. Overall, participants primarily used the system to close groups that they had currently open rather than opening groups they had previously saved (as can be seen on Table 5.2). One participant stated in the survey that what he found most useful was “creating my own groups for work-related topics (e.g., everything related to a JIRA ticket was in one group which made resuming work really fast)”, while three participants found a variation of “opening groups” to be the most useful. One participant mentioned an explicit hindrance to full adaption in that “what kept them from fully relying on the extension for task switching was that it would have required a significant change to their workflow” (Pa6). The focus mode was not used by any of the participants, which one participant reasons could be because “work is too fragmented” (Pi1). However, our survey also showed that two participants did not know about the focus mode feature at all, while three explicitly responded that they had not used it, and one gave a “Neutral” response regarding its usefulness.

Participant	Interactions	Opened Groups	Closed Groups
1	22	4 (18%)	18 (82%)
2	16	2 (13%)	14 (88%)
3	27	1 (4%)	26 (96%)
4	24	4 (17%)	20 (83%)
5	1	1 (100%)	-
6	34	10 (29%)	24 (71%)
Total	124	22 (18%)	102 (82%)

Table 5.2: Overview of User Interactions with Tab Groups. Includes interactions with both manually created groups and curated suggestions. *Source: Statistical analysis based on the interaction logs of our six primary participants.*

5.2 RQ2a: Can we accurately group browser tabs that are related based on knowledge workers' workflows?

As a first step in evaluating the quality of our automated group suggestions (i.e., our second research question), we looked at what participants thought of their composition and accuracy, what they thought of as good and bad examples of suggestions, and how well the suggestions matched their work in the browser.

Participants found the topics represented by our suggestions to be well-separated and internally cohesive. When asked about the quality of the contents of their suggested tab groups and how these could match their tasks, participants largely judged the groups to be well-formed in that they matched separate topics they had worked on. One participant stated that “initial suggestions matched past activities quite well”(Pa1), while another mentioned that “what is striking is that the topics are very consistent. These groups are all separate topics” (Pa3). Furthermore, participants also found that groups primarily contained tabs that are somehow similar or related. One participant stated that “inside of groups it is quite cohesive”(Pa4), while another mentioned the similarity in that “it often grouped things that were somehow similar” (Pa2), and a third found that “they rarely thought that something was completely misplaced” (Pa6). While the opinions on the accuracy of our suggestions are split according to our survey (as can be seen in Fig. 5.7), half of the participants agree that suggestions correspond to what they think of as a task. Furthermore, we found that only about 8% of suggestions that were discarded were marked as wrongly grouped, as will be shown in the next section in Table 5.4.

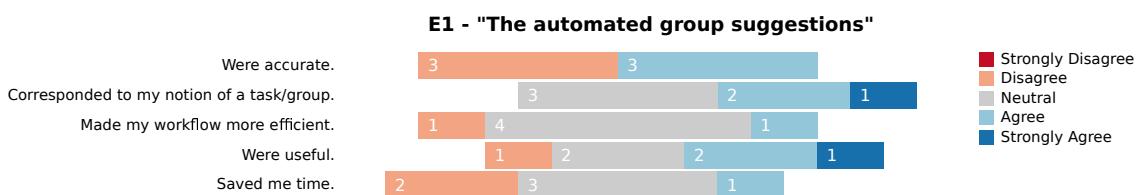


Figure 5.7: Follow-Up Survey Question E1: “The automated group suggestions”. *Source: Own depiction based on responses as given by our six primary participants.*

Participants identified challenges in the precision/recall trade-off of suggestions, as well as in missing semantic recognition. While participants largely found the groups to have high precision in terms of not containing tabs that do not belong, one participant mentioned that the groups were too small and that the approach trades off higher accuracy for lower coverage of the user's activities: "the groups were good but I think you traded a higher precision for a lower recall" (Pa6). Related observations were made by another participant, who mentioned that "there are three groups that contain similar things and that are that same topic in principle" (Pa4) or that "there were often things missing from the suggestions, so they were incomplete" (Pa5). Other observations were made in that the current approach is primarily based on temporal features (i.e., what was used before/after what), while some problems could be better handled by semantic features (e.g., comparing the titles of tabs). One participant explicitly mentioned that "what it does not yet manage is grouping things that are semantically similar" (Pa2) and elaborated that "it would be great if it grouped things with the same ticket name and number" (Pa2).

5.3 RQ2b: Can we use our identified tab groups to support knowledge work in the web browser?

When trying to support knowledge workers in their tasks using recommendations, a second factor is even more crucial than the accuracy of the recommendations: how useful and applicable the recommendations are to the current context of the user. While recommendations that are not accurate intrinsically have a low potential for further use, that does not mean that accurate recommendations are useful by default. The second part of our evaluation on the quality of suggestions presents the gathered qualitative feedback on the usefulness of our suggestions, as well as an analysis of how participants had interacted with our suggestions.

While participants liked the approach, they found the reusability of our suggestions to be limited. Participants identified a key issue in that the approach was providing them with accurate suggestions for past activities that they, however, would not perform again in the same scenario. One participant summarized that "the suggestions are less representative of the tasks I use all the time, and more of the things that I search for and once I have found it won't need it ever again" (Pa4). One participant stated similarly that "most often, I discarded groups because I would not use them again, not because they were not grouped well" (Pa1). The same participant mentioned that, in principle, accurate and useful suggestions "would provide an advantage when compared to bookmark folders" (Pa1). When surveyed about the percentage of our suggestions that participants would deem relevant for their work context, we received responses ranging from 15% to 50%, with a mean of 25% and a standard deviation of 12.65%. Furthermore, when we surveyed participants on the percentage of one-off tasks in our overall suggestions, we got widely varying responses between 10% and 90%, with a mean of 62.67% and a standard deviation of 27.9%.

Participants saved a few suggestions, and discarded many more. The most significant part of interactions across all participants were the suggested groups and tabs that they discarded, as summarized in Table 5.3. We expected that the number of discards could be significantly higher than the number of accepts, purely for the reason that users might be more selective with what they save vs. what they discard (e.g., saving only the most important vs. discarding all that is not useful). Overall, participants described their workflow with suggestions as having "received suggestions and sometimes just renamed and organized them" (Pa6) or having "accepted a 'work' suggestion and only had to rename it" (Pa1). Furthermore, one participant described cherry-picking from different suggestions to build or extend manual groups: "The automated suggestions contained

useful things that I sometimes cherry-picked and moved to my own groups" (Pa2). When surveyed about the percentage of our suggestions that they had interacted with (which we defined as having used them at least in part), participants responded within a range of 5% to 33% and with a mean of 14.67% and a standard deviation of 10.23%.

Participant	Interactions	Accepted		Discarded	
		Groups	Tabs	Groups	Tabs
1	4	1 (25%)	-	2 (50%)	1 (25%)
2	26	-	-	26 (100%)	-
3	24	4 (17%)	2 (8%)	10 (42%)	8 (33%)
4	15	3 (20%)	2 (13%)	6 (40%)	4 (27%)
5	22	1 (5%)	-	20 (91%)	1 (5%)
6	33	4 (12%)	1 (3%)	28 (85%)	-
Total	124	13 (10%)	5 (4%)	92 (74%)	14 (11%)

Table 5.3: Overview of User Interactions with Suggestions. Accepted Groups are Suggestions that have been saved in entirety, while Accepted Tabs are single tabs that have been dragged from suggestions to a curated group. Similarly, Discarded Groups are suggestions that have been fully discarded, while Discarded Tabs represent single tabs that have been removed. Source: Statistical analysis based on the interaction logs of our six primary participants.

The vast majority of discarded groups were labeled as not being useful, with some being discarded for unspecified reasons, and only a few discarded because of an inaccurate grouping. Based on an analysis of the discard reasons that had been provided by participants for each discarded group (as shown in Table 5.4), we found that 65.2% of discards were due to the suggestion not being useful, with only 7.6% being discarded for wrong or inaccurate grouping. 27.1% of discards did not have a specific reason, which could either be because the reason did not fit in the two other categories, or because the participant did not explicitly select a reason. Furthermore, the ratings associated with these discards mostly did not deviate from the default value (i.e., 3*), with 17 ratings below 3* and 16 above 3* and a corresponding mean of 2.96* (see Table 5.5). Examples for suggestions that would be discarded as wrong or not useful are shown in Fig. 5.8 and Fig. 5.9.

Reason	Count	%
Wrong	7	7.6%
Not Useful	60	65.2%
Other	25	27.1%
Total	92	

Table 5.4: Overview of Reasons provided when Discarding Suggestions. Source: Statistical analysis based on the interaction logs of our six primary participants.

Rating	Count	%
1*	8	8.8%
2*	9	9.9%
3*	58	63.7%
4*	11	12.0%
5*	5	5.5%
Total	91	

Table 5.5: Overview of Ratings given for Discarded Suggestions. Source: Statistical analysis based on the interaction logs of our six primary participants.

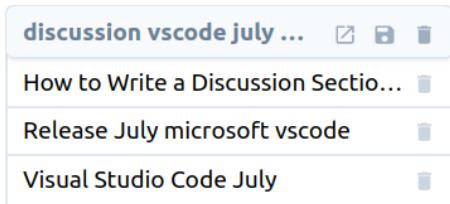


Figure 5.8: Example for a Wrongly Grouped Suggestion. The suggestion contains artifacts from multiple contexts, i.e., a *thesis writing* and a *release monitoring* context. The depicted grouping probably occurred because the tasks were worked on simultaneously, introducing switches between artifacts that do not belong together. *Source: Own depiction of a real example observed by researchers.*

A large part of the suggestions that were marked as not being useful contained tabs related to a specific search task or tabs related to a one-time activity like reading news and other articles. In one specifically bad example, a participant showed us a set of Instagram images that were grouped because they had been looked at sequentially or back and forth. Another example is articles in general, as “specific articles on Medium are things that you look for once, close it, and don’t reuse it again” (Pa4), and “it is bad when news articles are grouped after reading them” (Pa5). Additionally, some exemplary bad groups mentioned by participants contained “separate items from shopping” (Pi1) or “different hotels on booking.com” (Pi1), though the opinions on whether these kinds of groups would be useful or not were inconsistent, as other participants mentioned that “a group with different hotels is fine” (Pi2) and “I liked the suggestion that showed me the other furniture that I had looked at on IKEA” (Pa1). Fig. 5.9 visualizes a one-time search task.

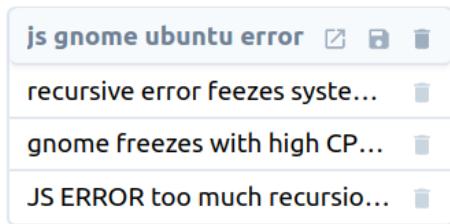


Figure 5.9: Example for a Suggestion representing a One-Time Search Task. The suggestion contains artifacts related to a search task for a very specific problem. While it is not wrongly grouped, it would probably not be needed again after the problem is solved. However, some participants found this useful in terms of having a historical view of their past problems. *Source: Own depiction of a real example observed by researchers.*

5.4 Overall Experience and Impact

In addition to our research questions, we evaluated the overall usability and impact of our extension to see: A. whether usability issues might have impacted our analysis and conclusions, and B. whether some of the auxiliary non-grouping related tab management features could have potential in supporting knowledge work. For this additional evaluation, we have performed a survey containing System Usability Scale (SUS) and functionality-related queries, as well as further analyzed some of the data we had collected.

5.4.1 Work with the Extension

To evaluate the impact our extension had on the efficiency and productivity of our participants, we surveyed them on a set of statements related to these areas. As shown in Fig. 5.10, while our system did not change their productivity or work fragmentation significantly, at least two participants found the system helpful in terms of task resumption.

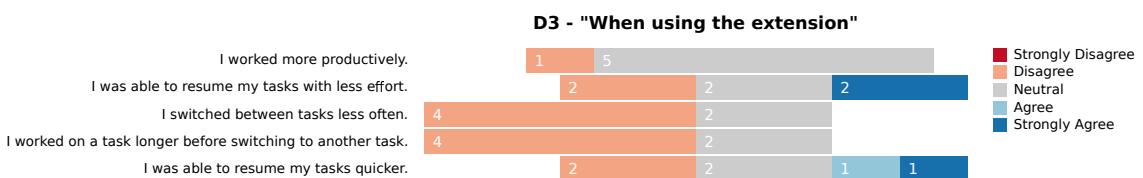


Figure 5.10: Follow-Up Survey Question D3: “When using the extension”. Source: Own depiction based on responses as given by our six primary participants.

Some participants applied the system to novel and unexpected use cases. While our system was primarily conceptualized to support task switching and resumption by enabling participants to persist suggestions for further use, some participants found use cases that worked around the limited reusability of our suggestions. When surveyed on the most useful functionality of the system, one participant mentioned “seeing an overview of the suggested tabs and groups, almost like a better bookmarking system where you are able to group and organize visited tabs”. A “historical” use case where “one can see the things that have been worked on or looked for, as in a history” (Pa2) was mentioned by three participants independently. One participant explicitly mentioned having preferred this use case to our conceptualized one: “I think I would use it more as a kind of history to review past activities, instead of resuming my task-based work” (Pa6). The same participant suggested that “the history approach is a different perspective on further development, which might make it useful to have a timeline instead of a list” (Pa6). Furthermore, the system was also described as “a kind of long-term memory to organize things I might need to look up again” (Pa6) by curating relevant artifacts from suggestions and through manual additions.

Participants found most features useful, especially the ones related to manual grouping. In our survey, we asked participants about their thoughts concerning specific features of our approach, and whether they found these features useful or not useful. As shown in Fig. 5.11, most features were judged to be useful by the majority of participants, though several features were unknown or not used by participants. On average, opening and closing interactions had the highest number of votes for their usefulness.

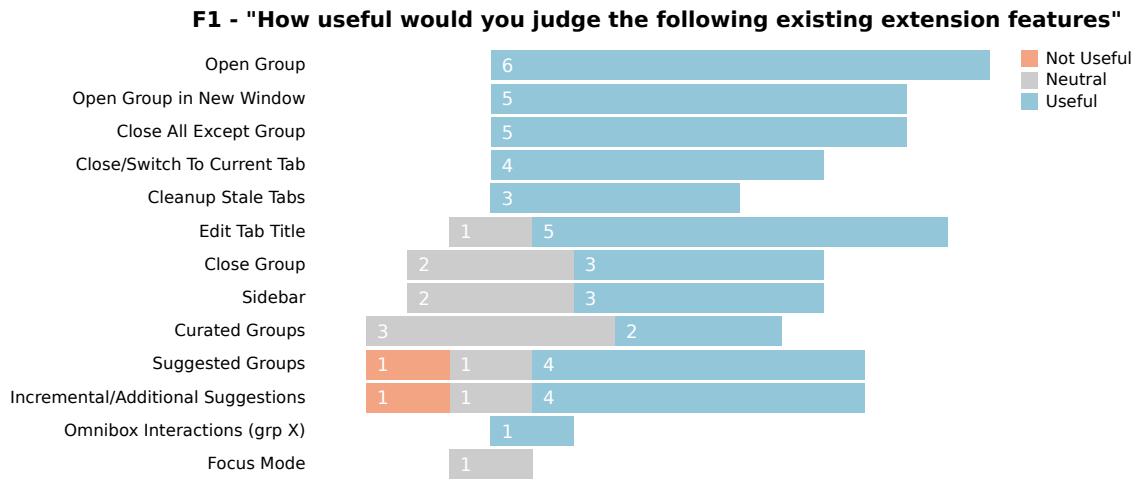


Figure 5.11: Follow-Up Survey Question F1: "How useful would you judge the following existing extension features". Source: Own depiction based on responses as given by our six primary participants.

Participants would like to have additional organizational capabilities to represent their different contexts. Besides asking participants about their opinion on existing functionality, we also evaluated what functionality they would like to see in further iterations of the tab grouping interface. As visualized in Fig. 5.12, the most-wanted feature was related to having a further level of organization for tab groups (i.e., higher-level groups of groups). One participant described this as: "I would like a feature for creating folders of groups, something like a higher-level context of groups" (Pa5). Other wished-for features include better integration with native browser features (i.e., bookmarks and synchronization), as well as functionality that improves the maintenance of artifacts (i.e., search and manual additions).

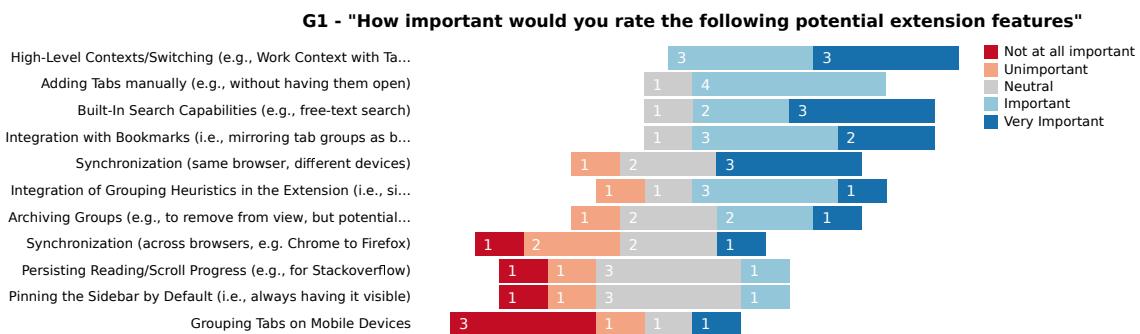


Figure 5.12: Follow-Up Survey Question G1: "How important would you rate the following potential extension features". Source: Own depiction based on responses as given by our six primary participants.

5.4.2 System Usability

We measured the usability of our system using the System Usability Scale (SUS), which is a method that is known for providing relatively reliable results with a lower level of effort when compared to other usability questionnaires (Brooke, 1996). We surveyed each participant on a set of 10 standard questions with response options on a Likert scale of agreement.

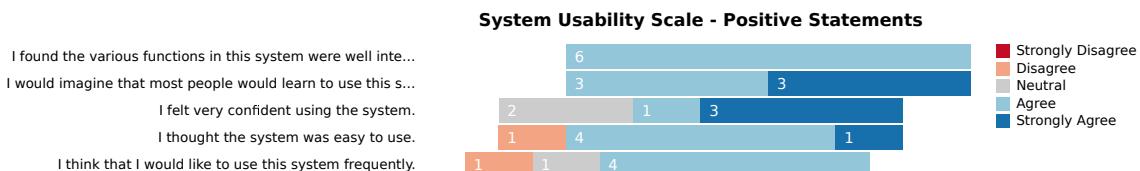


Figure 5.13: Follow-Up Survey Question B1: “Concerning the usability of the extension in general” (Positive Statements). Source: Own depiction based on responses as given by our six primary participants.

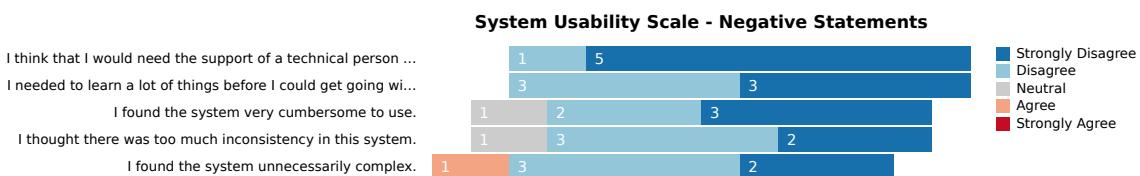


Figure 5.14: Follow-Up Survey Question B1: “Concerning the usability of the extension in general” (Negative Statements). Source: Own depiction based on responses as given by our six primary participants.

When evaluating the aggregated answers of participants as shown on Fig. 5.13 and Fig. 5.14, we can see that users found the system relatively easy to get started with. The majority of participants would like to continue using such a system. When we evaluate the aggregated SUS score for each participant (as shown on Fig. 5.15), we can see that the responses of all except one participant resulted in scores in a range of [78, 92], which we could call Good to Excellent based on relative comparisons with SUS benchmarks. The responses of the final participant yield a SUS score of 62, which would only be OK or Fair in terms of usability and certainly not the targeted result of any such system.

In our interviews, participants named a set of usability issues:

- **Hidden Functionalities:** The majority of participants thought it to be non-intuitive and not obvious that there would be a context menu with additional, and sometimes critical, features hidden in it. For example, the functionality to remove a curated group was only available in the context menu, which lead some participants to believe that the X used to close all tabs in a group would actually delete it. These participants thus mentioned their growing list of empty groups that they had been unable to remove. This backfired, as we had placed that functionality in the context menu specifically to prevent confusion between “closing” and “removing” and to reduce the clutter present in the User Interface (UI).

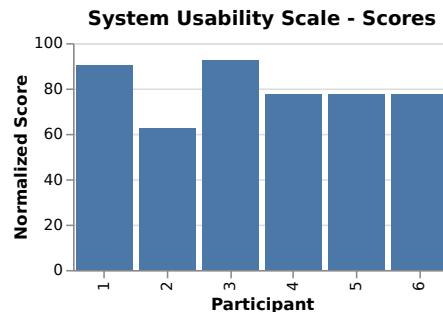


Figure 5.15: Follow-Up Survey Question B1: Normalized SUS Scores for all Participants. Source: Own depiction based on responses as given by our six primary participants.

- **Improved Visual Support:** One participant mentioned that the mostly textual interface requires a lot of cognitive energy to process: “Without visual support, I need to spend quite a lot of mental energy to distinguish groups” (Pa6). A suggestion was that “seeing the favicons or domain of a page” (Pa6) would be a helpful support.

5.4.3 Auxiliary Features (“Decluttering”)

In addition to features related to tab group curation and suggestions, we also provided users with a few options to clean up their browser environment. For example, we enabled them to easily close their open tabs from the list of “Current Tabs” in the main UI or sidebar. In contrast to the tab bar, this view always presents the title of the tabs so that a closing decision could be made more easily. Additionally, we implemented an approach for the detection and batch closing of tabs that had been open but unused for over one hour (which we refer to as “stale” tabs). Furthermore, having a less cluttered environment could also be a side effect of consistently applying the tab grouping approach for task switching, as the need to keep open tabs for later reference should be significantly lower.

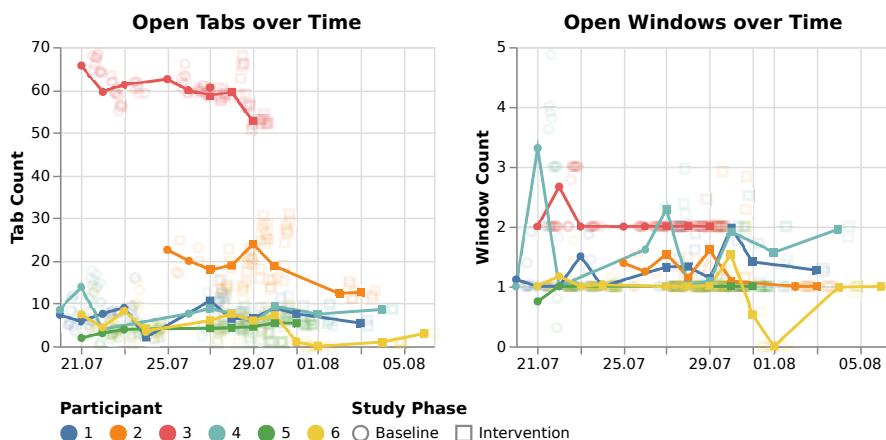


Figure 5.16: Development of Open Tabs and Windows over Time. Source: Own depiction based on the interaction logs of our six primary participants.

The number of concurrently open tabs did not change significantly. Neither did the portion of old or stale tabs. As the majority of our participants had described, one of the main problems in browser-based work lies in the steadily growing number of open tabs, but potentially no longer necessary. Participants “liked being able to close current tabs from the view” (Pa5) and “have closed old tabs from the extension view” (Pa5). However, while there were some downward trends for single participants (as shown in Fig. 5.16), there was no overarching significant pattern that could be observed in our statistical data. Furthermore, participants explicitly stated that they “did not have less tabs open because of the extension” (Pa5), and the majority disagreed with the statement that they “had fewer tabs open at a time” (as can be seen in Fig. 5.17). Additionally, our analysis did not show any consistent change in the distributions of tab age or staleness, also hinting that the clutter in the environments of participants did not necessarily change. While this goes in agreement with the observations that the overall tab count remained relatively stable, we would have hoped for a decrease in old or stale tabs.

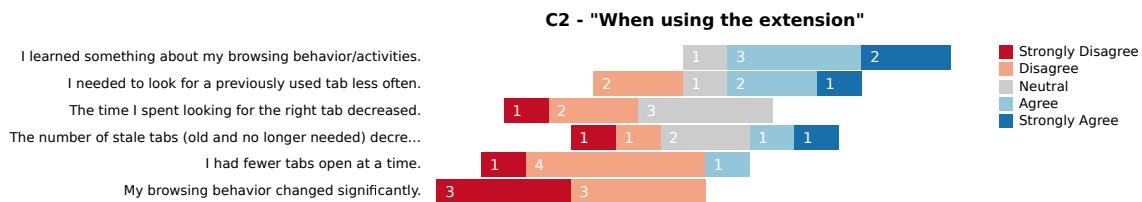


Figure 5.17: Follow-Up Survey Question C2: “When using the extension”. Source: Own depiction based on responses as given by our six primary participants.

Chapter 6

Threats and Limitations

The primary limitations of our work consist of the composition of the set of participants we have observed, as well as the length of the primary phases of our field study (i.e., two weeks in total).

Short and Inconsistent Study Time While we were able to gather initial data and feedback from our participants, we believe that our approach was partially unable to work to its full potential because of the limited study duration. Incorporating a tool into one's workflow requires a certain amount of time for learning and habit formation, and some participants explicitly mentioned that they would be fully ready to incorporate the tool after having gotten used to it during the intervention period. Furthermore, the phases of the study did not have the same length across participants, as scheduling constraints caused the initial setup and intermediate switch of phases to happen earlier or later. Additionally, the data of two participants were either incomplete towards the beginning or end, indicating that the prototype had not always gathered data correctly. This also limited our statistical analysis, as even a few days of missing data restricted our ability to draw statistical conclusions.

Homogenous Set of Participants Our sets of participants were homogenous in that we had two female pilot participants aged 30-32, who were knowledge workers in Finance, and six male software engineers/computer science students aged 26-29 during our primary study. While this convenient sample of participants does not provide an appropriate sample of the general population of knowledge workers, we initially felt that it would be reasonable for our initial exploration. However, as we have found in our study, browser work tends to differ between disciplines: while general knowledge workers like our pilot participants reported having many repetitive tasks they perform in the browser, the software engineers described often performing one-off search tasks for specific problems. This set of limitations also restricted the way our participants felt able to implement and work with our approach, and we conclude that it would be necessary to evaluate the approach on a more heterogeneous sample to evaluate its generalizability.

Chapter 7

Discussion

Based on the results and limitations as presented in preceding chapters, this chapter discusses both the findings from evaluating our tab grouping interface in general (Section 7.1), as well as the challenges faced in our automated grouping approach (Section 7.2). Furthermore, we relate our findings to previous research and summarize potential future work for both of these parts.

7.1 Tab Grouping Interface

As we have seen, our idea of tab grouping can be useful for task resumption and work organization given the right context. Participants in our study liked the overall principle of tab grouping and mentioned wanting to continue using such a tool to organize their work. Still, as we have seen earlier, our participants adopted only parts of the approach to their workflow. We primarily attribute this to factors like habit-forming and our strict definition of what constitutes a group, as discussed in more detail hereafter.

In general, knowledge workers have often formed habits in terms of working with a browser, so they need both an approach that can easily be integrated and does not get in the way of their workflow, as well as the willingness to adapt their workflow to incorporate it. While we tried to support as many different workflows as possible, there might be more guidance or functionality required to foster a thorough adaptation of tab grouping. For example, while we provided a way of focusing on a single task by “closing all tabs outside of a selected tab group”, such interactions might cause users to feel uneasy, as they could lose relevant information (i.e., tabs they left open for reference). We need to carefully think about the impact of the functionality on the user’s workflow, as well as what would additionally be needed to allow for full adaptation. In this specific example, we could add a way of restoring the closed tabs so that users do not risk losing any information.

In addition to missing functionality, the overall length of the study might have been too short to allow participants to fully adapt their workflow or habits. Participants explicitly commented that “I think I would now be ready to really work with the tool, as I have built my set of groups” (Pa6) and “The time I had was enough to get a first impression and create initial groups” (Pa1), meaning that a longer study could allow us to gather more in-use information on tab grouping in general (i.e., our first research question). Furthermore, a study could skew the results in that participants do not want to adapt their workflows only to have to resume their old way of working after its conclusion.

We had initially imagined a tab group as representative of a regularly repeating task with a well-defined set of artifacts. However, we have found that there are multiple types of tasks: on the one hand, there are different kinds of tasks in terms of how often they are repeated, as some tasks are performed once and never again (e.g., a one-off search), while other tasks are performed repeatedly

over an extended period (e.g., checking email). On the other hand, there are also multiple types of tasks in terms of their composition. Some tasks are well-separated and can be represented with a fixed set of artifacts, while other tasks consist primarily of artifacts that regularly change or that are not previously known to the user. For example, some tasks might be search tasks that evolve, and that the user does not get a clear picture of until they have been completed.

Each of the task types has different requirements in terms of tool support: a recurring task could be grouped either manually or automatically and using our interface, allowing for it to be resumed or restarted the next time it is needed. However, one-off tasks or evolving search tasks are both hard to recognize automatically, and cannot be reasonably supported with a group suggestion. Better support for such tasks could be provided with other (visual) cues that highlight their temporary nature and composition (e.g., by highlighting tabs in similar colors). Bernstein et al. (2008) approached this in an interesting way: artifacts that are related are drawn closer to each other on a large canvas, but there are no clear lines defining which artifacts constitute a group. Instead, the approach relies on the spatial understanding of users, which allows for artifacts to be in multiple groups, or no clear groups at all. Our findings also motivate more clearly why some approaches (e.g., Morris et al. (2008)) focused explicitly on search activities instead of trying to generalize to all types of tasks.

Additionally, different disciplines seem to have different needs regarding support for the types of tasks: we found that software engineers tend to have fewer clearly delineated tasks and often work on temporary tasks (e.g., tickets) or one-off searches. From interviews with our pilot participants, we conclude that other knowledge workers might have more tasks that are well-separated and regularly repeating. As our approach focused mainly on groups representing a regularly repeating and well-defined task, this could explain at least partially why our developer-oriented participants did not fully embrace the interface.

Our current interface is organized in the order of creation of curated groups (or acceptance of suggested groups) and contains primarily textual information. As shown in related work like Rattenbury and Canny (2007) and Bernstein et al. (2008), visualizations that encode information spatially (e.g., utilizing size and distance) or with other means can aid in comprehension of the overall structure. As elaborated previously, such approaches could also help us incorporate a less strict grouping approach and support it visually. Furthermore, we have received several feedbacks regarding extended visual support and the potential to lessen the cognitive load and reduce the time that users need to spend with our interface.

As a further extension of visual support mechanisms, restoring the latest reading state of artifacts (e.g., the scroll position on an article) upon resuming a task, as well as visualizing reading progress on the list of artifacts, could allow users to more quickly resume their work (Hahn et al., 2018). Furthermore, as shown by Leiva (2011), indicating the last interaction made with an artifact (e.g., the last position of the mouse before closing the artifact) also has a high potential of improving efficiency, though this would be more challenging to incorporate.

Besides visual support, several of our participants had expressed an interest in more automation for tab management and cleanup. While we currently only provide suggestions, further developments might include automated actions, such as opening related tabs when one tab of a group is opened, or automatically stashing stale tabs that are no longer needed. To prevent confusion or frustrations caused by such fully automated actions, any approach acting autonomously would need to work with the highest possible confidence.

7.2 Automated Tab Grouping

Properly maintaining an artifact management system (e.g., browser bookmarks or tab groups) can incur a significant manual workload. Supporting the maintenance of tab groups with automation can enable users to work more efficiently, as well as to learn more about their behaviors. While our initial approach to automated tab grouping produced relatively accurate suggestions in terms of what was grouped, we encountered significant issues in terms of the usefulness and relevance of our suggestions. Primarily, our suggestions turned out to not always be reusable, as they often matched tasks that had already been completed. The problem of reusability of recommendations was also observed in preceding research: participants in Rattenbury and Canny (2007) similarly noted that groupings were based on past activities, limiting their reusability.

We attribute a portion of these issues to a developer-specific workflow with many one-time search tasks. Therefore, we would expect improvements with prolonged usage of the approach, as the detection of regular tasks will improve over an extended period. However, besides recurring tasks, we have elaborated that there is an entire spectrum of tasks that could profit from tool support. Instead of trying to filter out one-time tasks, the approach might incorporate and support them in different ways. For example, a regular task could still be represented and suggested through a tab group, while one-time tasks could be supported with automation or (visual) cues that highlight their temporary nature and composition (e.g., by highlighting tabs in similar colors).

The community detection algorithm that we apply is, in principle, capable of detecting complex community structures. For example, communities can be nested into hierarchies, meaning that a larger community comprises several smaller communities. When applied to the problem of tab grouping, this could allow us to not only detect tab groups for separate tasks but also to detect what tasks are related and form an overarching context (e.g., “work”). Many participants had mentioned missing a higher-level context to separately represent sets of groups (e.g., contexts for personal or work activities). Deriving such higher-level contexts, in addition to tab groups, could allow us to support this use case not only manually, but with automated measures.

Additionally, similar to related work on automated task assignment, we could extend our approach to automatically detect which task the user is currently working on (i.e., which tab group he is working in). This could improve the relevance of our suggestions (e.g., focus on tab groups that are related to the current task/group and show these groups in a fish-eye view) and support further parts of the user’s workflow (e.g., support a full process of related tasks, rather than having the user choose everything on their own).

Finally, our approach currently relies primarily on the temporal ordering of tab visits (i.e., X was visited before Y, Y was switched to from X), though some semantic similarity features are built-into the reweighting procedures (e.g., to reduce weights of links where URLs are very similar). A further development might include more significant semantic heuristics in terms of comparing metadata or full-body contents of web pages to predict edges in the graph data structure, or to process suggestions before or after they are processed with temporal heuristics. Related work has often incorporated a set of heuristics in an ensemble and largely achieved improved results.

Chapter 8

Conclusion

Browser work takes on an increasingly important role for knowledge workers: more and more services are built for the web and become highly integrated with other services and platforms. However, browsers are not yet optimal vessels for task-based workflows that incorporate many artifacts, as browsers treat all artifacts as independent entities, not accounting for possible relationships regarding tasks or workflows. Furthermore, the artifact management capabilities of browsers (i.e., tabs, bookmarks, and history) are severely constrained, leading to browsers often becoming cluttered and hard to efficiently navigate, which creates an additional opportunity for distraction.

In our thesis work, we have designed and developed an artifact grouping approach for the browser consisting of two key parts: a tab grouping interface allowing for more fluid interactions when compared to bookmarks and folders thereof, and a heuristics engine that automatically suggests tab groups based on past browser behavior. We have evaluated our approach in a two-week field study with six participants and found that tab grouping in general, as well as with our interface, is well-liked and wished for. Our findings indicate that participants tend to have problems with the clutter in their environment and apply various strategies to handle said clutter, but would generally like an approach like tab grouping to reduce their maintenance overhead. While we found that our prototype is currently not applicable to the workflows and tasks of all users, the adaptation of our approach to supporting other types of tasks (e.g., one-time tasks) could improve applicability.

Our attempt at an approach to reduce the overhead of tab grouping, automated tab group suggestions, was found to be relatively accurate in terms of what was grouped. Participants noted that suggestions matched topics that they had worked on well, but also concluded that groups were sometimes not complete in terms of containing everything related to a topic. Furthermore, the suggestions were often not reusable for future work, as they matched past one-time tasks. While a part of this issue could be attributed to our study setup and the workflows of software engineers, a large part stems from issues in distinguishing between long-term and short-term tasks.

We see a lot of potential for extensions based on our approach, especially regarding improvements of suggestion and automation capabilities, but also in terms of supporting manual artifact organization. When implemented alongside extended support for other task types, the approach could be of use to a broad set of users.

Bibliography

- Aula, A., N. Jhaveri, and M. Käki (2005). Information search and re-access strategies of experienced web users. In *Proceedings of the 14th international conference on World Wide Web*, pp. 583–592.
- Bernstein, M. S., J. Shrager, and T. Winograd (2008). Taskposé: exploring fluid boundaries in an associative window visualization. In *Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08*, Monterey, CA, USA, pp. 231. ACM Press.
- Bohlin, L., D. Edler, A. Lancichinetti, and M. Rosvall (2014). Community Detection and Visualization of Networks with the Map Equation Framework. In Y. Ding, R. Rousseau, and D. Wolfram (Eds.), *Measuring Scholarly Impact*, pp. 3–34. Cham: Springer International Publishing.
- Brooke, J. (1996). Sus: a quick and dirty'usability. *Usability evaluation in industry*, 189.
- Cockburn, A. and B. McKenzie (2001). What do web users do? an empirical analysis of web use. *International Journal of human-computer studies* 54(6), 903–922.
- Dragunov, A. N., T. G. Dietterich, K. Johnsrude, M. McLaughlin, L. Li, and J. L. Herlocker (2005). TaskTracer: a desktop environment to support multi-tasking knowledge workers. In *Proceedings of the 10th international conference on Intelligent user interfaces - IUI '05*, San Diego, California, USA, pp. 75. ACM Press.
- Esmailian, P. and M. Jalili (2015, November). Community Detection in Signed Networks: the Role of Negative ties in Different Scales. *Scientific Reports* 5(1), 14339.
- Fortunato, S. (2010, February). Community detection in graphs. *Physics Reports* 486(3-5), 75–174. arXiv: 0906.0612.
- Hahn, N., J. C. Chang, and A. Kittur (2018). Bento Browser: Complex Mobile Search Without Tabs. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*, Montreal QC, Canada, pp. 1–12. ACM Press.
- Henderson, D. A. and S. Card (1986, July). Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Transactions on Graphics (TOG)* 5(3), 211–243.
- Huang, J. and R. W. White (2010). Parallel browsing behavior on the web. In *Proceedings of the 21st ACM conference on Hypertext and hypermedia - HT '10*, Toronto, Ontario, Canada, pp. 13. ACM Press.
- Jones, W., H. Bruce, and S. Dumais (2003). How do people get back to information on the web? how can they do it better? In *Interact.*

- Kersten, M. and G. C. Murphy (2006). Using task context to improve programmer productivity. In *Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering - SIGSOFT '06/FSE-14*, Portland, Oregon, USA, pp. 1. ACM Press.
- Leiva, L. A. (2011). MouseHints: easing task switching in parallel browsing. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems - CHI EA '11*, Vancouver, BC, Canada, pp. 1957. ACM Press.
- Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, Volume 10, pp. 707–710.
- Li, P. L. (2016). *What makes a great software engineer*. Ph. D. thesis.
- Maalej, W., M. Ellmann, and R. Robbes (2017, June). Using contexts similarity to predict relationships between tasks. *Journal of Systems and Software* 128, 267–284.
- MacKay, B. and C. Watters (2008). Exploring Multi-session Web Tasks. pp. 10.
- MacKay, B. and C. Watters (2009). Building support for multi-session tasks. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems - CHI EA '09*, Boston, MA, USA, pp. 4273. ACM Press.
- Mark, G., V. M. Gonzalez, and J. Harris (2005). No task left behind? examining the nature of fragmented work. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 321–330.
- Mark, G., S. Iqbal, M. Czerwinski, and P. Johns (2015). Focused, aroused, but so distractible: Temporal perspectives on multitasking and communications. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pp. 903–916.
- Meyer, A. N., T. Fritz, G. C. Murphy, and T. Zimmermann (2014). Software developers' perceptions of productivity. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 19–29.
- Miles, M. B. and A. M. Huberman (1994). *Qualitative data analysis: An expanded sourcebook*. sage.
- Morris, D., M. Ringel Morris, and G. Venolia (2008). SearchBar: a search-centric web history for task resumption and information re-finding. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, Florence, Italy, pp. 1207. ACM Press.
- O'Conaill, B. and D. Frohlich (1995). Timespace in the workplace: Dealing with interruptions. In *Conference companion on Human factors in computing systems*, pp. 262–263.
- Oliver, N., M. Czerwinski, G. Smith, and K. Roomp (2008). RelAltTab: assisting users in switching windows. In *Proceedings of the 13th international conference on Intelligent user interfaces - IUI '08*, Gran Canaria, Spain, pp. 385. ACM Press.
- Oliver, N., G. Smith, C. Thakkar, and A. C. Surendran (2006). SWISH: semantic analysis of window titles and switching history. In *Proceedings of the 11th international conference on Intelligent user interfaces - IUI '06*, Sydney, Australia, pp. 194. ACM Press.
- Page, L., S. Brin, R. Motwani, and T. Winograd (1999, November). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

- Pilzer, J., R. Rosenast, A. N. Meyer, E. M. Huang, and T. Fritz (2020, April). Supporting Software Developers' Focused Work on Window-Based Desktops. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, Honolulu HI USA, pp. 1–13. ACM.
- Rajamanickam, M. R., R. MacKenzie, B. Lam, and T. Su (2010). A task-focused approach to support sharing and interruption recovery in web browsers. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems - CHI EA '10*, Atlanta, Georgia, USA, pp. 4345. ACM Press.
- Rattenbury, T. and J. Canny (2007). CAAD: an automatic task support system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '07*, San Jose, California, USA, pp. 687–696. ACM Press.
- Rosvall, M., D. Axelsson, and C. T. Bergstrom (2009, November). The map equation. *The European Physical Journal Special Topics* 178(1), 13–23.
- Sahm, A. and W. Maalej (2010). Switch! Recommending Artifacts Needed Next Based on Personal and Shared Context. pp. 12.
- Shen, J., J. Irvine, M. Goodman, S. Kolibaba, A. Tran, F. Carl, B. Kirschner, S. Stumpf, and T. G. Dietterich (2009, February). Detecting and Correcting User Activity Switches: Algorithms and Interfaces. pp. 9.
- Stumpf, S. (2005, July). Predicting User Tasks: I Know What You're Doing! pp. 6.

Appendices

A. Architecture

The overall architecture of our application consists of two critical parts: a browser extension that can be installed in standard-compatible browsers (e.g., Chrome, Firefox, Edge), and a background application that runs on the JVM on the local machine. These two components communicate through the Native Messaging standard defined by browser vendors (i.e., an API based on message exchange).

Browser Extension

The browser extension is a Typescript application based on the React frontend framework, making it very flexible in terms of component reusability. We developed the extension with Typescript for improved type-safety, as we interact with a lot of browser APIs that have clearly defined parameters and responses. The overall layout is based on Tailwind CSS and Material UI.

When the extension is loaded by the browser environment, it prepares a set of listeners for relevant browser events. For example, the extension would subsequently be notified on every new tab that is created, or every time the user goes into an idle state. Most of this information is not directly used by the extension but instead piped through to the heuristics engine for further processing.

Heuristics Engine

The JVM backend for the tab grouping extension has been built with Scala, allowing it to run cross-platform on the JVM while being able to use the language features and functional libraries of Scala. We use the Akka framework to optimize our application in terms of parallel processing, as browsers themselves can use a lot of resources and our background application should not slow down user devices unnecessarily.

The Akka framework is based on the “Actor Model”, which describes the principle of splitting an application into many independent parts (i.e., actors) that are decoupled and communicate exclusively using inter-actor messaging with clearly defined (typed) message types. The framework allows for high flexibility in terms of parallel processing and scaling, and can be used to build highly resilient systems that are distributed across many machines.

We package our application to native binaries using sbt-native-packager. This allows us to provide binaries for Linux and Mac OS, as well as for Windows in conjunction with the Wix packaging framework.

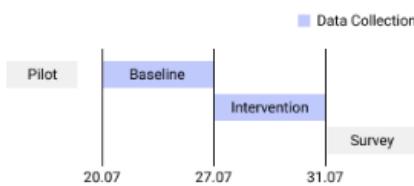
B. Extension: Tutorial

Introduction

Welcome and thanks for your participation in to the user study on Automated Tab Organization.

You should have received and signed a consent form with the full study procedures; nevertheless, the following provides a short summary of the study procedures.

Phases



Baseline Phase (1 work week)

You setup a heuristics backend (next tutorial step) that will analyze your browsing behavior and collect aggregate statistics about tab interactions. The collected data will allow us to compare the effect of our grouping functionality with a baseline.

Intervention Phase (1 work week)

You enter an activation key that enables manual and automated grouping features. You can use these features while going about your normal workweek. We will prompt you to interact with our suggestions at least once a day (e.g., discard a subjectively bad group).

Study Conclusion

To conclude your participation in the study after the second phase, you will be asked to participate in an interview and fill out a survey about your participation. Additionally, you will be able to review the data that we have collected before you can send it to us.

Interview (ca. 20 min)

We will use the interview to discuss the quality of our suggestions using some of your real examples. We will record the interview for the purposes of transcription if you explicitly agree on the consent form.

Survey (ca. 10 min)

The survey will collect some demographic information as well as more standardized data and feedback about the study.

Data Submission (ca. 5 - 30min)

You will be able to review and censor all of the data that we have collected and stored on your local machine.

We will provide you with a set of supporting scripts that should make this much easier.

Heuristics Setup

The next step of the tutorial will take you through the necessary setup steps for the heuristics backend that will collect data and provide you with suggestions.

[NEXT STEP](#)

1 Introduction

2 Heuristics Setup

3 Data Collection

Figure 1: Step 1 of the Initial Tutorial. The first step of the tutorial contains a short repetition of the study procedures. *Source: Screenshot of the tutorial embedded in the browser extension.*

The screenshot shows a web-based tutorial for setting up the Heuristics Engine on Linux. At the top, it says "Heuristics Setup on Linux". Below that, it asks to open a new Terminal session and run the command "pwd". It notes that this will need to be executed in the Terminal throughout the installation process. A section titled "System Requirements" is expanded, stating that certain system requirements must be met before installation. It specifically mentions Java 11 Runtime Environment and provides commands to update the system and install Java via apt-get. It also includes a command to check Java's version. The output of these commands is shown as "OpenJDK Runtime Environment (AdoptOpenJDK)". There is a "CONTINUE" button at the bottom of this section. Below this, there are two more sections: "Heuristics Setup" and "Heuristics Connection", each with an unchecked checkbox. At the very bottom, there are navigation links: "PREVIOUS STEP" (Introduction), "NEXT STEP" (Data Collection), and a numbered progress bar showing "1 Introduction", "2 Heuristics Setup" (which is highlighted with a blue circle and a checkmark), and "3 Data Collection".

Figure 2: Step 2 of the Initial Tutorial. The second step of the tutorial guides users through the installation process for the heuristics engine. Depending on their browser and operating system, appropriate step-by-step instructions will be shown. *Source: Screenshot of the tutorial embedded in the browser extension.*

Data Collection

Now that you have installed and enabled the heuristics engine, we will start tracking your browsing behavior in terms of interactions and aggregate statistics.

Collected Data

We will not (cannot) track any interactions performed in incognito mode or in browsers other than this one.

We collect the following data while you are browsing with tracking enabled:

Phase 1 & 2

- Aggregate Statistics (e.g., number of tab switches, number of open tabs, average age of tabs)
- Tab interactions (e.g., opening a new tab, closing a tab, updating the location of a tab, tab switches)
- Usage statistics (i.e., active minutes)
- Suggested tab groups (periodic snapshots of our algorithm output)

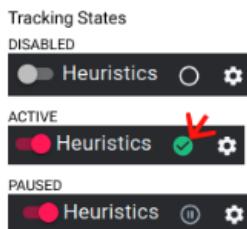
The collected data about tab interactions contains the URL and Title of the respective tabs but without the potentially sensitive URL parameters at the end of the URL.

Phase 2

- Heuristics interactions (e.g., accepting a suggested tab group, discarding a suggested tab)

Pause & Resume

You can pause and resume tracking of your behavior by clicking on the (pause/green) circle as shown below:



Data Storage and Review

All data is stored in your user home directory (i.e., `/home/X/tabs` on Linux and `C:/Users/X/tabs` on Windows). You are free to have a look and censor the collected data at any time; however, we would advise you to focus on the final review after the conclusion of the second phase. This will save you time and allows you to work more efficiently using our review and cleanup tooling.

PREVIOUS STEP

NEXT STEP



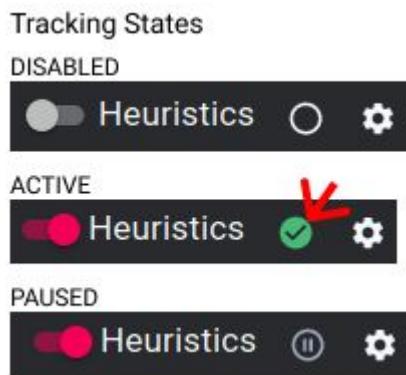
Figure 3: Step 3 of the Initial Tutorial. The third step of the tutorial shortly repeats the most critical information regarding data collection. *Source: Screenshot of the tutorial embedded in the browser extension.*

C. Extension: Feature Documentation

Feature Overview

Pause & Resume

You can pause and resume tracking of your behavior by clicking on the (pause/green) circle as shown below:



Keyboard Shortcuts

- `Ctrl+Shift+Q`: Open the extension sidebar on any page
- `Ctrl+Shift+S`: Open the extension overview in a new tab

You can customize these shortcuts in the Browser extension settings (Chrome: `<chrome://extensions/shortcuts>`, Firefox:

<https://support.mozilla.org/en-US/kb/manage-extension-shortcuts-firefox>).

Manual Grouping

Current Tabs

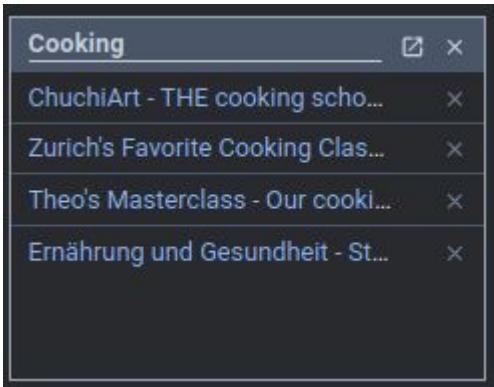
Current Tabs	
Window 1	✖
Project Plan	✖
Tab Groups	✖
Study Procedures – Figma	✖

The list of current tabs allows you to close any currently open tab (click on the 'x').

Additionally, tabs that have been open but unused for more than an hour will be marked as stale. You can close all of these tabs by clicking on the cleanup button in the top right.

You can drag and drop tabs to any other curated group that you might already have, as well as to the plus to create a new group.

Curated Groups



You can open and close an entire group (i.e., all of its tabs) with the buttons in the top right. Grouped tabs that are currently open will be marked and can be closed by clicking on the respective 'x'. Opening the context menu of a grouped tab (right-click on the tab) will further allow you to remove the tab from the group and to open it in a new tab.

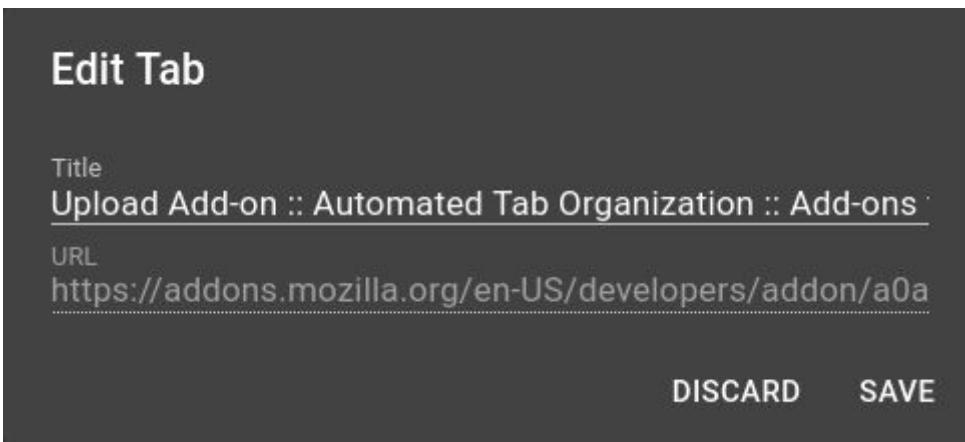
The context menu of a group (right-click on the header) additionally allows you to delete the group, close all tabs **except** the ones in that group, as well as to open that group in a new window.

Focus Mode



We have also implemented a `Focus Mode` to improve your workflow if you regularly switch groups/tasks. Once focus mode is activated and you open a tab group (as described above), all other tab groups that are currently open will be closed. This allows you to switch tasks while cleaning up unneeded tabs simultaneously.

Edit Tab



You can edit the title of a tab using the pencil icon on the respective tab. The URL cannot be changed; instead, open a tab with the other URL and drag it to that group.

Sidebar

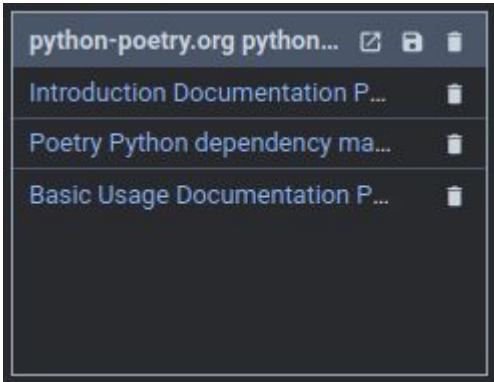


In addition to the view you can open with the extension button in your browser bar, we also offer a sidebar view that you can open on any webpage you are on. To do so, click on the sidebar trigger button in the bottom left of your page. You can also use the keyboard shortcut `Ctrl+Shift+Q` (or `Cmd+Shift+Q` on MacOS) to directly open the sidebar.

The sidebar allows you all of the same interactions that you know from the other extension view, while bringing these interactions closer to where you are browsing.

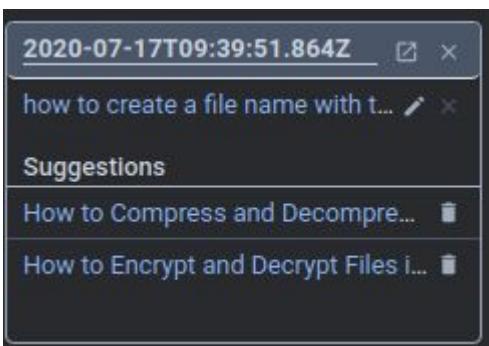
Suggestions

Once the grouping heuristics have collected enough data, you will be presented with suggestions for groups that you might want to add to your curated groups. Such a suggestion might look as follows:



When you want to persist that suggestion in your curated list, click the save button in the top right. If the suggestion is not helpful, you can discard it using the trash icon. You can also discard single tabs within that suggestion and, after doing so, persist only what remains. Additionally, you can freely drag and drop from suggested groups to your curated groups.

When a significant overlap is detected, the heuristics might also suggest additions to existing curated groups. These could look as pictured below:



You can accept these additions by dragging the additional tab into your curated group, or discard them by clicking the trash icon. When you discard a group, you will be asked for a reason so that we can learn more about the performance of our grouping heuristics.

How appropriate was this suggestion?

:(:-(:(:) :()

Why are you discarding it?

WRONGLY GROUPED

NOT USEFUL AT THIS TIME

OTHER REASON

DO IT!

A modal dialog box asking for feedback on a suggestion. It includes a rating scale from sad to happy faces, a question about discarding reasons, and three specific options: 'WRONGLY GROUPED', 'NOT USEFUL AT THIS TIME', and 'OTHER REASON'. A large blue 'DO IT!' button at the bottom is intended for saving the discarded suggestion.

D. Interviews: Guiding Questions

Intermediate Interview (after Baseline Phase)

- Ask intermediate questions
- Enable grouping features and check whether everything works (including initial suggestions)
- Evaluate initial suggestions and existing data collection (maybe try other tuning)
- Explain accepting groups, tabs, and discarding groups, tabs
- Suggest installation of new tab [addon](#)

How do people work in a browser and what problems do they encounter?

Goal: get an intuition for the types of work people do and how they go about that work

- How much of your working time is spent in a browser?
- How fragmented is your work in a regular situation (e.g., in terms of focus on a task and in terms of interruptions)? Do you use “do-not-disturb” mode or other tools/approaches to block distractions?
 - “Focus” dimension: “working on everything at once” vs. “focusing on a single task”
 - “Interruption” dimension: “interrupted all the time” vs. “focus for prolonged periods”
- How do you use bookmarks and the browser history while working in the browser?
 - No explicit bookmarks, everything searched for when needed (history or search)
 - Bookmarks for the most important pages (e.g., Gmail, Calendar)
 - Bookmarks in folders
 - Organized by task (e.g., CRM, Coding)
 - Organized by context (e.g., Google Services)
 - Organized by project (e.g., Platform for Bank X)
 - Organized by time/frequency (e.g., Daily, End of Month or Morning/Evening)
 - Organized differently or in a more complex structure
- What kinds of (recurring) tasks do you generally work on in a browser?
 - High-level contexts like “marketing”, “project management”, or “email”
 - Low-level tasks like “interpreting google analytics” or “managing issues for X”
- How do you manage tabs in general: close after use, keep open for reference/reminder, keep open forever, reset once a day, ...? How many tabs do you have open at a time?
- How do you manage the tabs/websites that belong to a task?
 - Do you organize the tabs in separate browser windows? Multiple windows side-by-side?
Distributed across desktops?
 - Do you often (accidentally) switch to open tabs that are not relevant to your task?
 - What strategy do you use for recurring tasks or tasks that you plan to return to?
 - Reopen everything once it is needed (e.g., from bookmarks or search)
 - Keep browser windows/tabs open to be able to return to them (& reminder)
 - Move tabs to another window/desktop to reduce distraction
- What are hindrances that you regularly encounter while working in a browser? How do you try to cope with them?
 - Losing overview or being distracted due to too many open tabs
 - Daily cleanup/closing the browser
 - Unable to find something important to return to (that was previously open)

What do people think about the initial suggestions?

- What are your first impressions? Do these suggestions match your view of a tab groups/task?
- Do you think you will be able to profit from suggestions like these in the coming week?

- Why & how or why not?
- What are obvious issues that you can see?

Follow-Up Interview (after Intervention Phase)

- Send download link for Python tooling
 - https://tabs.fra1.digitaloceanspaces.com/submission/2020-08-04_submission.zip
- Ask follow-up questions
- Step through timeline of generated groups
 - Install graphviz on local machine
 - Install issues on MacOS: <https://github.com/pygraphviz/pygraphviz/issues/100>
- Explain and run data postprocessing tool
- Thank for participation and explain conclusion procedures (survey and data submission)

How was participants' overall experience?

Goal: figure out whether there have been issues that prevented our approach from being useful to its full potential, and whether people liked the overall experience.

- How would you describe your **overall experience** with our browser extension and the study?
 - Was the approach helpful for you overall and if so, how?
 - What did you like or dislike about the approach for the extension and study?
 - How well were you able to incorporate the extension into your daily workflow?
 - E.g., in terms of supporting task switches and resumption
 - Did you have to adapt your workflow?
- How would you describe the **user experience** in general?
 - Were there specific problems that prevented you from using the extension or participating in the study as you wanted/expected?

How well does our approach solve the problems participants have?

Goal: figure out whether what we built actually helped them accomplish their tasks better. Also, find out whether our approach fit their mental model in terms of tasks/grouping (maybe they know more about that than in the intermediate interview, now that they have been thinking about/"in" groups for a while, e.g. they might know whether they like temporal groups more than other ones)

- What types of groupings did you interact with? Why or why not?
 - How useful do you consider the different grouping features: manual vs. automated and why?
- Did the concept of grouping tabs generally resonate with your idea of a task in the browser?
 - If yes, how so? If no, what would be a better representation of a "task"?
 - What were tasks that you have worked on repeatedly over the past two weeks?
 - Can you describe a set of tabs that would perfectly match your view of such a task?
- Did your browsing behavior change when using the extension?
 - E.g., did you have less tabs open at a time?
- Have you tried working in Focus Mode (i.e., on one task at a time)?
 - If yes, what was your experience?
 - If no, did you know about the feature?
- How would you describe the accuracy of the automated suggestions?
 -> to support this question, go through the timeline of suggestions with them (~5 minutes)
 -> supported by the Python script generating a set of images from the saved grouping snapshots

- Can you name a few bad examples? What was bad about them (wrong/not_useful/other)?
- Can you name a few good examples? What was good about them?
- Were there suggestions that you actually ended up using? Which ones?

How could we further improve our approach?

Goal: figure out what future work could improve our results.

- Would you recommend such a tool to a colleague? Why or why not?
- What additional features or extensions would you need to fully rely on such an extension?
 - I.e., to replace folders of bookmarks, or similar, if used
 - I.e., to rely on tab groups in terms of switching tasks by opening and closing groups

E. Survey: Questions



This survey represents the final part of our user study on Automated Tab Organization.

It should take you about 15 minutes to complete.

Thank you for your significant contribution to our research!

Section A: Demographics

A1. How old are you?

A2. What is your profession?

• **1**

A3. How would you describe your role at your workplace?

10. The following table summarizes the results of the study.

A4. How many years of working experience do you have?

Section B: General Usability

B1. Concerning the usability of the extension in general:

I think that I would like to use this system frequently.

I found the system unnecessarily complex.

I thought the system was easy to use.

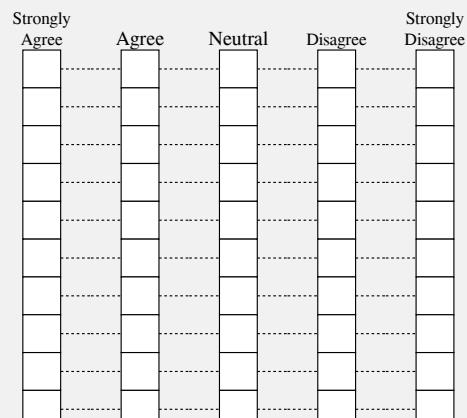
I think that I would need the support of a technical person to be
I found the various functions in this system were well integrated.

I thought there was too much inconsistency in this system.
I would imagine that most people would learn to use this system

I found the system very cumbersome to use.

I felt very confident using the system.

I needed to learn a lot of things before I could get going with this





Section C: General Browsing Behavior

C1.

How many browser windows do you usually have open at a time (estimate)?

<input type="checkbox"/>											
--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------

How many browser tabs do you usually have open at a time (estimate)?

<input type="checkbox"/>											
--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------

How much of your time is spent on regular/repeated tasks (e.g., scheduling) vs. one-off/search tasks (e.g., looking for a specific solution on SO) (percentage estimate)?

<input type="checkbox"/>											
--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------

C2. When using the extension:

My browsing behavior changed significantly.

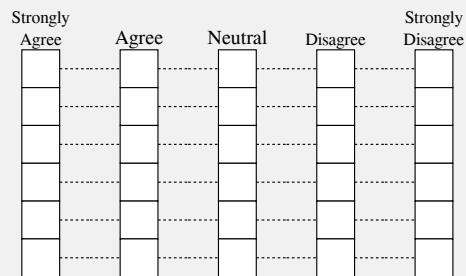
I had fewer tabs open at a time.

The time I spent looking for the right tab decreased.

The number of stale tabs (old and no longer needed) decreased.

I needed to look for a previously used tab less often.

I learned something about my browsing behavior/activities.



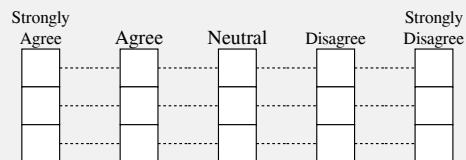
Section D: Tab Grouping

D1. When I am working:

I am regularly interrupted.

I regularly achieve a state of Flow (tunnel-vision, full focus).

I work on a number of projects/tasks simultaneously.

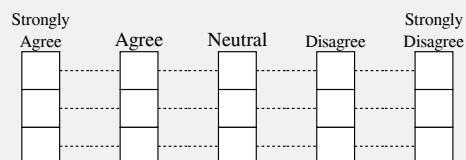


D2. The concept of grouping tabs in general (irrespective of our implementation):

Can improve productivity significantly.

Fits my way of working.

Should be better supported by browsers.



D3. When using the extension:

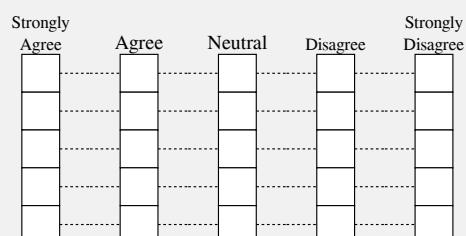
I switched between tasks less often.

I worked more productively.

I was able to resume my tasks quicker.

I was able to resume my tasks with less effort.

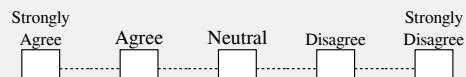
I worked on a task longer before switching to another task.



Section E: Group Suggestions

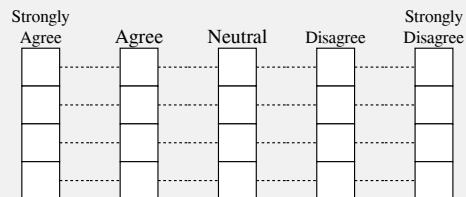
E1. The automated group suggestions:

Made my workflow more efficient.





- Saved me time.
- Corresponded to my notion of a task/group.
- Were accurate.
- Were useful.



E2. When you think about your interactions with our suggestions:

What percentage of our suggestions were relevant to your work (estimate)?

What percentage of our suggestions did you interact with (i.e., use at least partially) (estimate)?

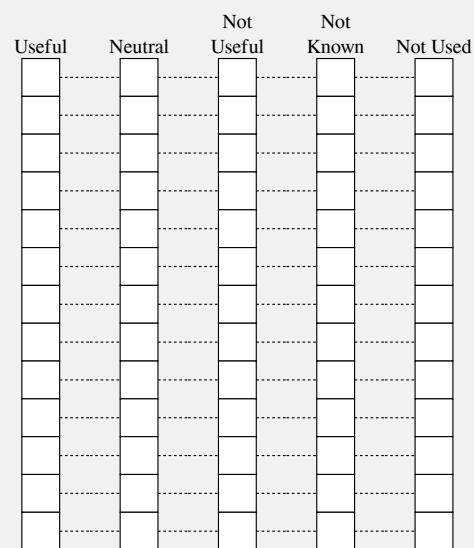
What percentage of our suggestions represented one-off tasks (i.e., tasks that you would not perform again) (estimate)?

E3. How many tab groups or suggestions did you create/accept and regularly interact with (estimate)?

Section F: Extension Functionality

F1. How useful would you judge the following existing extension features:

- Close/Switch To Current Tab
- Open Group
- Open Group in New Window
- Curated Groups
- Suggested Groups
- Incremental/Additional Suggestions
- Close Group
- Close All Except Group
- Sidebar
- Focus Mode
- Cleanup Stale Tabs
- Omnibox Interactions (grp X)
- Edit Tab Title



F2. What was the functionality that you got the most use out of during the study?

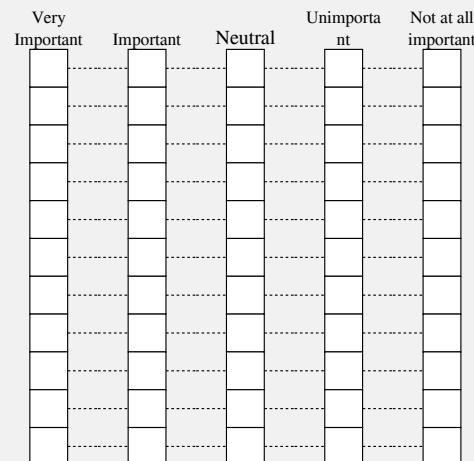
•



Section G: Future Work

G1. How important would you rate the following potential extension features:

- High-Level Contexts/Switching (e.g., Work Context with Tab Synchronization (same browser, different devices))
- Synchronization (across browsers, e.g. Chrome/Firefox)
- Grouping Tabs on Mobile Devices
- Persisting Reading/Scroll Progress (e.g., for StackOverflow)
- Adding Tabs manually (e.g., without having them open)
- Pinning the Sidebar by Default (i.e., always having it visible)
- Archiving Groups (e.g., to remove from view, but potentially
- Built-In Search Capabilities (e.g., free-text search)
- Integration with Bookmarks (i.e., mirroring tab groups as
- Integration of Grouping Heuristics in the Extension (i.e., simpler



G2. Are there any features not mentioned yet that would make the extension a better fit for your workflow?

G3. Is there any other feedback that you would like to share with us?

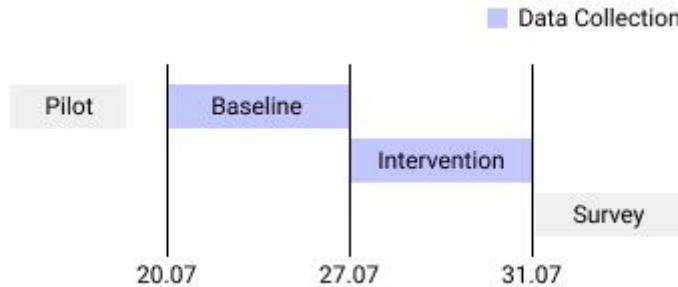
F. User Study: Welcome Document

Welcome!

Thanks for participating in our user study. By now, you should have received and signed a consent form with a summary of the study procedure. This document provides an overview of the most important steps and resources throughout the study, as well as a documentation of the features in the second part. You will find additional information in our in-browser tutorials and documentation.

Overview

The user study is split into two main phases:



Baseline Phase (1 work week)

You setup a heuristics backend that will analyze your browsing behavior and collect aggregate statistics about tab interactions. The collected data will allow us to compare the effect of our grouping functionality with this baseline.

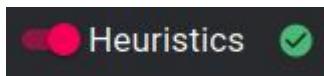
Intervention Phase (1 work week)

You enter an activation key that enables manual and automated grouping features. You can use these features while going about your normal workweek. We will prompt you to interact with our suggestions at least once a day.

Day 1 - Getting Started

To install the extension, simply open the link (see below) for your respective browser. After installation and a restart of your browser, you will be guided through the setup process for the heuristics backend (which is essential for suggesting groups). Please make sure that the heuristics backend is enabled at all times, except for times when you want to pause it explicitly. If the tutorial does not open even when you restart your browser, please refer to the tutorial link for your respective browser.

Once the setup has been completed successfully, heuristics should be enabled and the status display in the extension UI should show the following:



In case of installation problems, please contact me ASAP, as we cannot collect any data when the setup did not work.

Chrome

Important: You need to be signed in with your Google Account to download Chrome extensions.

Extension (required):

<https://chrome.google.com/webstore/detail/automated-tab-organizatio/jgcfdfaeaoogamdkfbdnfhkohibdjkd>

Tutorial (required):

<chrome-extension://jgcfdfaeaoogamdkfbdnfhkohibdjkd/tutorial.html>

If you use Chrome, you may additionally install the “New Tab” extension, which replaces your normal “New Tab” page with an overview of your tab groups. You should wait with this step until the second week of the study, as the grouping features will become available by then.

New Tab (optional):

https://chrome.google.com/webstore/detail/automated-tab-organizatio/npfoadapkhhheogjopeoldchoc_kpbkcd

Firefox

Extension (required):

https://tabs.fra1.digitaloceanspaces.com/webextension/automated_tab_organization-0.4.1-fx.xpi

Day 6 - Intervention Phase

At the start of the intervention phase, you will be sent an activation key that you can enter to enable tab grouping features (including both manual groupings and automated suggestions). You are encouraged to try out the features of the extension and to interact with the groupings, incorporating them into your normal workflows (i.e., to create groups that support your tasks, use focus mode, discard suggestions, etc.). The extension will remind you to interact with the suggestions at least once a day (if you are working in your browser).

Day 10 - Conclusion

Once the study concludes after 10 working days, you will be sent further information on the procedures for data submission. We will provide you with tooling to review, clean, and encrypt your data such that you can be comfortable with what you send us for analysis.

Feedback

If you have suggestions for improvements or features that you would like to see in such an extension, please add them to the form at <https://forms.gle/Tde9G6jQxzF8aX7b8>. Extension development will continue after the study concludes, so any such feedback is very welcome.

Email: roland.schlaefli@uzh.ch