Contextinator: Recreating the context lost amid information fragmentation on the web

Ankit Ahuja

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science & Applications

Manuel A. Pérez-Quiñones, Chair
Andrea L. Kavanaugh
Stephen H. Edwards

May 3, 2013
Blacksburg, Virginia

Keywords: Information Fragmentation, Activity-based Computing, Personal Information Management, Tool Integration, Web-based Information Systems, Contextinator

Contextinator: Recreating the context lost amid information fragmentation on the web

Ankit Ahuja

(ABSTRACT)

The web browser has emerged as a central workspace for information workers, where they make use of cloud-based applications to access their information. While this solution nicely supports access to their data from multiple devices, it presents a nightmare for organizing and coordinating data between tools for a single project. Information is typically scattered between various online tools, where storage and organization structures are replicated. Information workers are interrupted and have to switch between projects frequently. Once interrupted, resuming work on a project can be hard. To address this information fragmentation and the impact of work interruptions, I created *Contextinator*, a personal information manager for the web browser that lets information workers organize their work activity and information into projects. *Contextinator* assists in coordinating information for projects, thereby ameliorating information fragmentation for projects that live on the cloud. It assists information workers in context switching and resuming work after interruptions. In my thesis, I describe the problem of information fragmentation in the cloud. I discuss the different areas of related work of Personal Information Management, the design of *Contextinator* and how it is grounded in previous research. I briefly discuss how *Contextinator* is implemented. I then present the results from my field-evaluation of *Contextinator*. Finally, I conclude by discussing future work in this research.

Acknowledgments

It has been an incredible journey having Dr. Manuel A. Pérez-Quiñones as my advisor. His *wacky* ideas have been a constant source of excitement and motivation, and have encouraged me to think out of the box. His support and flexibility have helped me make it through some of the tougher times. His joyful nature has taught me not to take life too seriously and to keep moving forward. I will cherish my experience working with him throughout the rest of my career, and I hope to continue to collaborate with him in the future.

I have enjoyed working closely with Dr. Andrea Kavanaugh for the past two years. I thank her for her flexibility and guidance. The *Programming Languages* course given by Dr. Edwards is one of the most enjoyable courses I've taken in graduate school. I thank him for being a part of my committee.

My brothers, Sameer Ahuja and Anshul Dhir, have been a source of inspiration and guidance throughout my professional career. Thanks to Sameer's prior work at CHCI, I've always felt a part of a family here.

I thank Michael, Ben, Robert, Bert, Kumbie, Nai Ching, Sheriff and Yao for their ideas and support as I worked towards my Masters.

Finally, I thank my parents for giving me the opportunity to study abroad and learn and grow as an individual.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The web browser has emerged as a central workspace for information workers. Much of the information that these workers process resides on the cloud and is often accessed through individual web applications (e.g., Dropbox for files, Evernote for notes, Gmail for email). Since data for each of these applications resides in a device agnostic remote storage, this cloud-based approach is nicely suited for accessing data from multiple devices.

The emergence of powerful classes of portable devices such as tablets and smart phones, which support the increased mobility of information workers, has extended this trend. Advances in web technologies have made it possible for web applications to match the functionality and usability of desktop based applications. Unfortunately, these new technologies proliferate the fragmentation of information across collections.

The Personal Information Management (PIM) research community has already identified this as a problem [17, 18, 16]. The continuously growing number of applications for storing and managing different kinds of personal information create silos of data that are not interoperable with each other. There is no connection between the data stored in these applications, and the user has to individually manipulate and organize the data in each of them. For example, the labels in Gmail are completely independent of the folders in Google Drive and there is no structural connection between them. Applications provide very limited integration with other applications at best (e.g. downloading a file from an email in Gmail directly to Google Drive).

1.1 The Problem

Information fragmentation, which was already occurring on the user's desktop, is now occurring across information collections on the cloud. Information is hidden behind individual application interfaces and organizational structures for each of these collections, which pre-

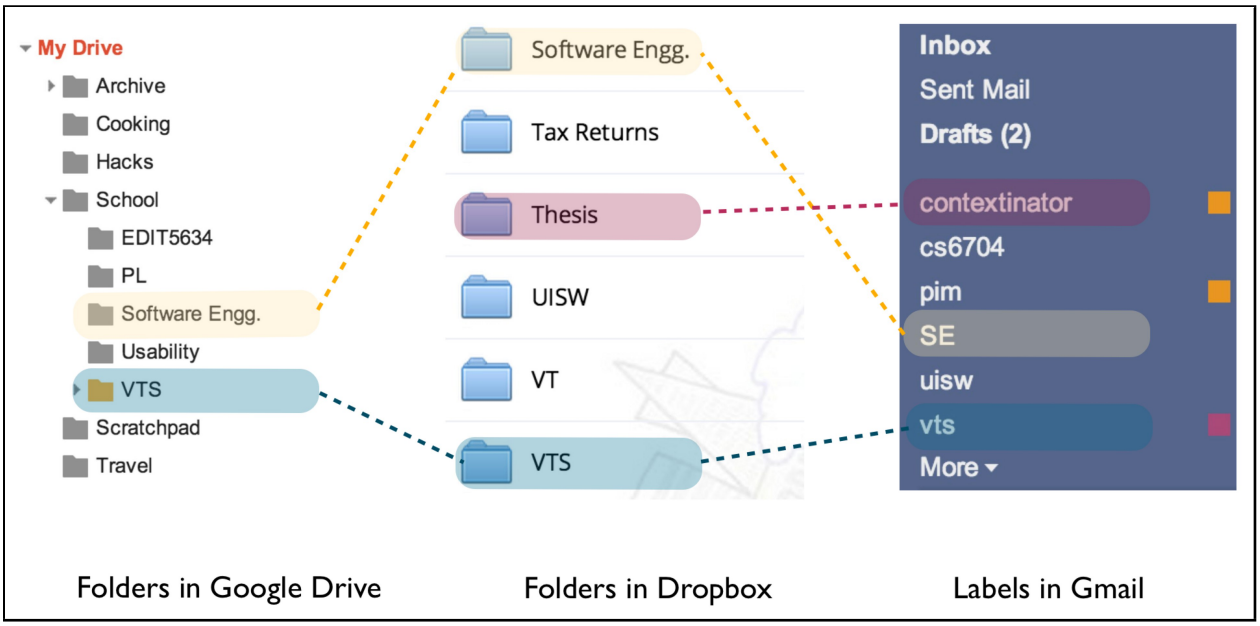

Figure 1.1: Fragmentation of information across collections.

vents information workers from taking a holistic approach to manage their information. Each of the cloud based applications defines its own *organizational unit* (e.g., label, folder, notebook) and *structure*. Even though one might have a label for a project in Gmail and a folder for the same project in Google Drive, there is no connection between the two and the user has to independently think of the two *organizational structures*. This fragmentation causes further problems when the user has to refind information for a project in any of these applications. Each of them has their own user interface and the user has to navigate each structure independently to access information for the same project.

This information fragmentation occurs due to the existence of *organizational units and structures* for the same project across different collections [18]. Figure 1.1 shows a subset of the Google Drive folders, Dropbox folders and Gmail labels for a typical user (based on my own and my research group's use of these tools). This user has *organizational units* for the same projects across these three collections, with slightly different labeling and organization. For example, "Software Engg.", "Software Engg." and "SE" correspond to the same project. While working on this project, the user may need to access information from any of these disparate collections, requiring traversal of that collection. In this example, the three collections slightly differ in their *organizational structure* (the "Software Engg." folder in Google Drive is located under the "School" folder, whereas the *organizational units* in the other two collections are at the root level). In some cases, only some collections may contain information for a project. For example, "Thesis" and "contextinator" correspond to the same project and only exist in Dropbox and Gmail respectively.

In addition to the problem of information fragmentation, information work is also fragmented, where information workers are interrupted and have to switch between projects frequently. Once interrupted, resuming work on a project can be hard. This has also been previously identified as a problem by the PIM community [20, 25, 37].

Switching between different applications to access information related to a project results in opening, traversing and closing of different *organizational structures* in these applications. This can make users prone to distractions as they see information unrelated to their current work. A user may see information pertaining to pending work from another project and get distracted. For example, a user who visits Gmail to reply to an email related to her current project, might see unread emails related to other projects and get distracted. Information workers need a way to capture and keep track of things they need to do and a way to prioritize them to enable them to accomplish important tasks. I designed *Contextinator* based on findings in the state of the art in this research domain to address these problems.

1.2 Approach

In order to tackle the problem of information fragmentation in the cloud, an approach is needed that enables information workers to manage, access and view the same project's information across different collections as part of a single *organizational unit*. In *Contextinator*, this *organizational unit* is a project. *Contextinator* lets users define a project as a collection of individual *organizational units* in different information collections. It then makes use of this context to enable easier access to a project's information across different collections. Figure 1.2 shows how the individual *organizational units* in different collections are collected together as part of a project in *Contextinator*.

To apply the concept of projects to all of the user's work, I designed *Contextinator* to let information workers organize all their work activity and information within the the web browser into projects. With *Contextinator*, users can group their web sessions and cloud-based artifacts (e.g., email, tasks, bookmarks, windows) into projects and manage them within the context of a project. *Contextinator* enables users to identify an *information view* for a project in different web applications (e.g. associating a label in Gmail with a specific project) to enable easier access to the project's information. A *Project Homepage* in *Contextinator* clusters together all information related to the project. I also created a task management system within *Contextinator* to help users capture and accomplish tasks for their projects.

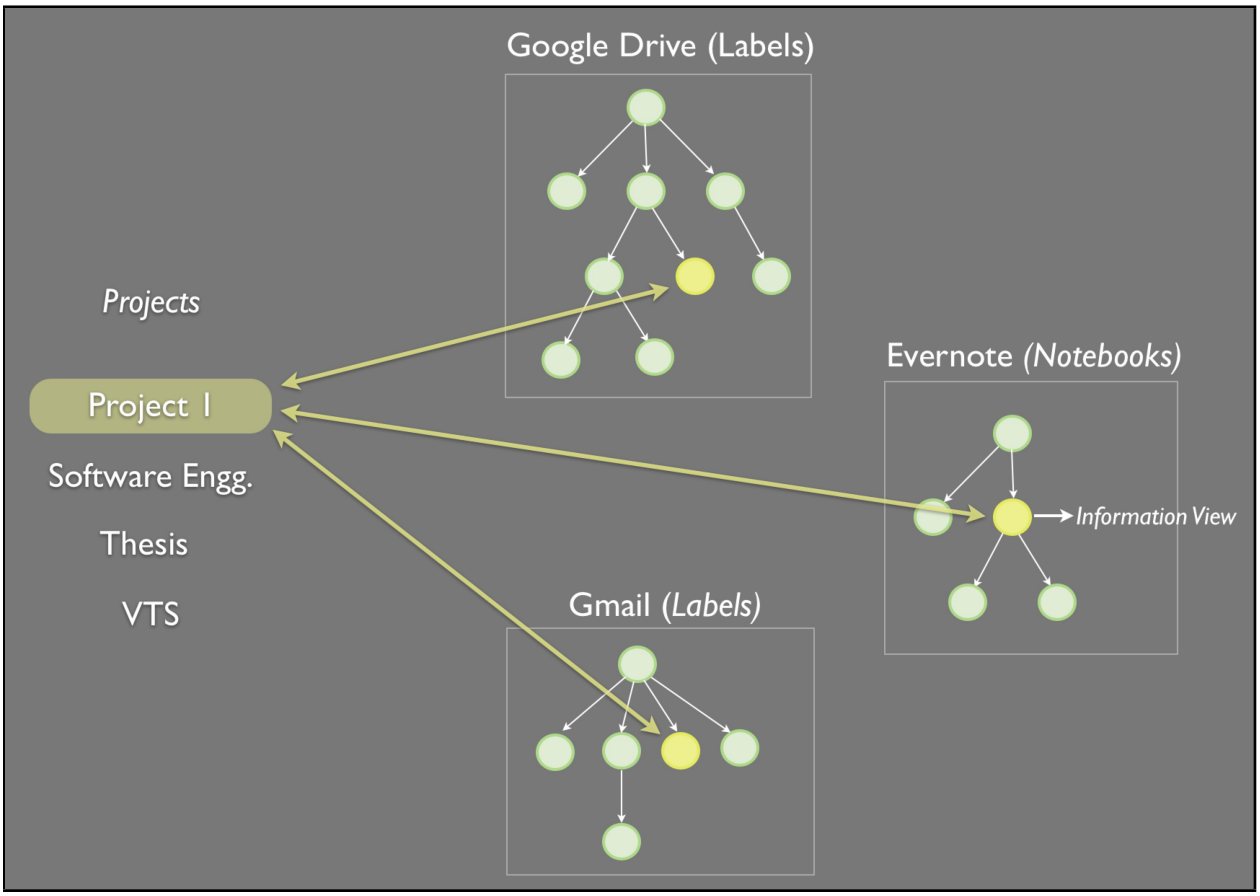

Figure 1.2: Information views in *Contextinator*.

1.3 Evaluation

Evaluating a PIM tool can be challenging [34]. Personal information is unique to each individual and users manage their own information collections using different strategies. It is difficult to design evaluations to capture all such collections and strategies completely, which makes it hard to study the usage of such a tool in a controlled environment. Users may be heavily invested in their existing tools and strategies for managing their information, and introducing a new tool into their workflow can be intrusive. I thus used a combination of naturalistic, field-test and case study approaches to evaluate *Contextinator*. I deployed *Contextinator* to participants recruited using a snowball sampling technique. I instrumented *Contextinator* to log user actions, and followed that up with interviews and an online survey. I then analyzed the data collected from these sources to gain insight into how users adopted the tool in their day-to-day work. Some of the questions I wanted to answer as part of my evaluation are:

- How do people split their work activity into projects?
- What are the advantages of clustering together project information?
- Can *information views* make it easier to access a project's information?
- How often do people switch between projects?

Through my evaluation of *Contextinator*, I saw some evidence that supports the design of *information views*, even though their usage was limited. Participants generally grasped the concept of projects easily and liked organizing their information into projects. The most well adopted features of the tool were the ability to switch between projects while working and restoring the state of a project's workspace. The task management system in *Contextinator* received positive responses from the few participants that actively used it. Some of the features of the tool received limited use, which could be partially due to the demographics of the participants for my evaluation, a lot of whom did not have a lot of information or projects they needed to manage. I feel that *Contextinator* has the potential to recreate the context lost amid information fragmentation in today's web-based tools.

1.4 Organization of Thesis

This thesis is organized as follows:

- In Chapter 2, I discuss the different areas of related work of PIM, including Information Fragmentation, Task Management, Activity-based computing and Multitasking.

- In Chapter 3, I discuss how the different design decisions for *Contextinator* are grounded in previous research.
- In Chapter 4, I present the implementation of *Contextinator*.
- I explain the methods I adopted for data collection and analysis in Chapter 5.
- This discussion is followed by the results from my field-evaluation of *Contextinator* in Chapter 6.
- I then summarize the findings from the evaluation and discuss implications for the design of future systems similar to *Contextinator* in Chapter 7.
- Finally, I conclude with future work in this research in Chapter 8.

Chapter 2

Prior Work

Several areas in PIM are related to this research. In particular, my work is related to Information Fragmentation, Task Management, Activity-based Computing, Multitasking, and other broader areas of user interface design such as Window Management. This chapter summarizes the relevant literature in these areas and presents how prior work has influenced the design of *Contextinator*.

2.1 Information Fragmentation

Information fragmentation occurs when our personal information is scattered over different devices, storage systems, and online tools, each with its own organizational structure. It is a 'pervasive problem in personal information management' [33].

The current trend in tablets and smart phones is for each application to have its own storage and its own data, with little to no integration or synchronization across the different types of artifacts. This trend results in the creation of information silos for each application. In order to cope with these different silos, information workers have had to develop different methods for organizing project related information. These can include using multiple folder hierarchies to organize documents related to projects [31], using a special folder (or tag) in an email client to hold messages related to a project [25], or using virtual spaces to separate windows of different projects [27].

A problem with this approach is that users end up maintaining duplicate organizational hierarchies. For example, users tend to have similar structures in their file system as they do in their email archives. These are difficult to maintain and to keep synchronized. Additionally, users are often faced with the question of where a document is located, as an email attachment or in a project folder in a file manager.

Several research efforts have endeavored to link these parallel structures. Boardman et al.

[18, 17] studied users' PIM organization strategies across different tools and identified some of the problems caused by information fragmentation, including:

- Management of certain types of information is compartmentalized between distinct tools.
- Coordinating production activities across different tools is difficult.
- Inconsistencies exist between different implementations of equivalent functionality.

To solve the second problem, Boardman et al. created a prototype to mirror folder structures between different PIM tools. Their users found sharing categories between tools intuitive and compelling. Their solution was only applicable to information collections stored locally (including the local file system, MS Outlook and locally stored bookmarks).

Bergman et al. [16] approached the problem as project fragmentation, where information was fragmented into different collections without relation to the common activity uniting them. Project fragmentation results in users having to store and retrieve their project related information artifacts in and from various locations with no structural connection. Their proposed solution was to use a single hierarchy to store all files of different formats under the same folder. Similarly, Jones et al. [29] suggested the development of a common structure that could be shared and manipulated by any number of tools. However, this requires all apps to manipulate a shared common structure, which will be a significant undertaking given the current dependence on individual data stores. While existing approaches have come up with different solutions to the problem of information fragmentation, they have failed to properly address the problem of fragmentation for information stored in the cloud.

This integration of information collections is also occurring in the commercial and open source tool space. Several tools address aspects of the information fragmentation problem to different levels of success. Unfortunately, most of these attempts try to minimize information fragmentation by creating tools to assist users to access information from different collections, without creating any structural link between the different collections. For example, CloudMagic [2] creates a unified search box to do keyword-based searches for information across all information collections. Attachments.me [1] enables access to Dropbox files while creating attachments in Gmail. At the time of this writing, Yahoo! announced that it would support the use of Dropbox for storage of attachments in their mail program, duplicating the behavior that Attachments.me provide.

2.2 Refinding

Refinding refers to the process of finding information that has been seen before. Capra and Pérez-Quiñones [19] found that refinding is a two-stage process, first focusing on relocating

the source of the information (searching) and then, a second stage of re-locating the specific information being sought (browsing). In the context of my work, the first step of relocating the source of information is identifying which collection the information is stored in. The next step is to search for that information and browse to refind the information within that collection. The different collections, as shown in Figure 1.1, make the refinding process difficult because the user has to remember upfront which collection has the information. This can be especially difficult when multiple collections may store the same type of information (e.g. files may be stored in Dropbox and as attachments in Gmail) [44]. Some tools in the commercial space, such as CloudMagic, try to make refinding easier by letting users search for information across all collections. However, these tools only have limited support to browse information. In my system, I have addressed refinding of information for a project by eliminating the need to browse independent information collections to locate the information for the project.

2.3 Task Management

Another area of research related to my work is Task Management. Information workers typically have a list of pending actions for each project. Bellotti et al. [12] studied task management to support the design of a task list manager. Tasks within each project can help information workers prioritize and maintain their attention over different projects [12, 25]. Tasks, especially when they appear in the way and in an always visible spot of the working space, can act as good reminders [14, 12, 25]. Bellotti also suggested that a task manager should support informal priority lists, to ensure near-term execution of priority actions. Prior work has also explored connections between email and tasks. Ducheneaut et al. [23] conducted a field study to explore the unanticipated uses of email resulting from the increased use of email as a personal information management tool. One of the things they found was that users often use email for information management functions such as to-dos, by marking up and resending oneself messages. Bellotti et al. [13] created a prototype called Taskmaster, that recasted email as task management and embedded task-centric resources in the email client. Email clients are also emerging in the commercial space, where email is being recast as a task management system [8, 7]. The suggestions from Bellotti defined the design of the task management system in *Contextinator*. I created the ability to flag tasks to support informal priority lists, and the tasks are shown in several places within the system to support the always visible and in the way properties.

2.3.1 Quick Capture

Information workers often capture things to do from the use of their other tools. An example is when a user is browsing the web and reading material online for pleasure. She finds an article that is relevant to a project she is working on and wants to discuss it with her

coworkers. She quick captures the webpage URL with a todo "Discuss with team at next meeting".

This kind of association can make accomplishing the task easier at a later time, as the user will not have to search for that webpage, and in turn, the webpage provides context for the task. In the user-subjective approach to PIM systems [15], Bergman suggested that capturing the context of an information item during interaction with it will help the user recall the information when it is next needed. Jones et al. [30] implemented quick capture as part of the personal project planner, where users could create rich text project plans and reference documents, email messages, web pages, etc. Users could drag and drop resources into the plan to create a link back to the original source.

Hanrahan et al. [26] added a quick capture ability within the email client to move information to wikis. The quick capture method in Mail2Wiki provided the worker with the ability to do a light categorization while they were still engaged with the context of the email. So that later, a deeper curation of the information could be done with the context of the previous curation.

Quick capture also exists as a feature in some commercial Getting Things Done (GTD) tools, such as OmniFocus, Things (with Quick entry), and RememberTheMilk (with Quick add). A common implementation of quick capture tools is as a browser or OS extension to capture a task quickly from anywhere (without switching applications). Another common implementation is email integration (ability to create tasks by sending email).

The existing work in quick capture inspired the feature to capture the context of a task in *Contextinator*'s task management system. In *Contextinator*, the captured information is categorized by associating the task with the currently active project. Users can capture a task's context in the form of a website URL (which could be a document, web page, email message, etc.) and notes. The URL and notes are shown alongside the task to enable recall and ability to reference the information later.

2.4 Multitasking and Interruptions

One of the typical characteristics of today's information workers is that they are often interrupted in the middle of their work with communication mechanisms (e.g., phone calls, text messages, email, etc.). Not all of these interruptions are about the same project, thus workers are constantly switching projects and multitasking.

Czerwinski et al. [20] did a diary study of the activities of information workers to characterize how people interleave multiple tasks amidst interruptions. They found that information workers switch tasks a significant number of times during a week, with an average of 50 shifts over the week. The returned-to projects were more complex, significantly lengthier in duration (twice as long as shorter-term projects), were interrupted more and were also rated

significantly harder to return to than shorter-term projects.

González and Mark [25, 37] found that people's information work is highly fragmented, where they spend an average of three minutes on a task before switching tasks and an average of 12 minutes on a project before switching to another project. They also found several ways in which people try to manage their information to handle constantly switching between different activities, including aggregating a project's different types of information into a single artifact.

González and Mark's work was instrumental in defining some of the features in *Contextinator*. To support multitasking and resumption of work after an interruption, I implemented quick switching between projects, save and restore of project state and aggregation of project information in *Contextinator*.

2.5 Project Organization

2.5.1 Activity-based Computing

Research in activity-based computing explores how to find a better mapping of real life projects to computing systems. People have been found to be generally skilled in defining projects, where each project consists of information including email, tasks, calendar, bookmarks, notes, documents and other artifacts. González and Mark [25, 37] introduced the concept of working spheres to explain how information workers conceptualize and organize their basic units of work. They claim that support for such organization in computing systems can improve the productivity and multitasking capability of information workers. Balakrishnan et al. [11] studied workers of Lotus Activities, an activity-centric computing (ACC) system, to see how workers aggregated activity-related artifacts together while using other collaboration tools. They found that to reduce fragmentation, ACC systems need to create consolidated access to activity artifacts.

A number of prototypes have been developed to explore this activity-based paradigm. Lifestreams [24] displayed a time-ordered stream of documents to replace the conventional files and directories, and users could go back in time to look at the documents they worked on. TaskTracer [21] monitored users' activities and resources accessed by the user while working on a particular task, and enables the user to access there records and restore task contexts. Kaptelinin [32] took a similar approach in UMEA, which tagged resources (documents, folders, URL and contacts) accessed by the user to the currently active project. Voida et al. [45] created an activity-based system where they linked organization of application windows and documents of a project by associating files saved on the desktop of a virtual space with the currently active project. This was the most actively adopted feature of their prototype. Users found it to be easier than filing since they did not have to manually find the project folder to store the files in.

The concept of working spheres and existing activity-based systems motivated the organization of information and work activity into projects in *Contextinator*. Some features in *Contextinator* are similar to some of the existing prototypes. The support for quick switching between projects is similar to the one in Gionarta and the *Project Homepage* in *Contextinator* is similar to the project window in UMEA, as both make project information visible in a single view.

2.5.2 Window Management

Window Management strategies enable people to better manage their workspace (desktop) on the computer. Prior work has explored grouping of related application and document windows to let information workers better organize their workspace.

Rooms [27] first introduced the concept of virtual desktops, which is now a part of window management systems of modern operating systems. Task Gallery [41] was a 3D window manager for task management. Kimura [35] extended the virtual desktop to make use of project peripheral displays to support background activities. GroupBar [42] extended the existing Windows Taskbar to enable grouping of windows into higher-level tasks and easy task switching. WindowScape [43] used implicit grouping for windows to let users organize and retrieve windows according to task.

Better management of space and sessions has also been explored specifically for the web browser, given the popularity of web applications for accomplishing common tasks, combined with the limited browser support for resumption of tasks. Rajamanickam et al. [40] created a task-focused web browser, where webpages could be grouped into tasks. Morris et al. [38] created SearchBar, a tool that stored users' search query and browsing histories, to support task resumption across multiple sessions. Wang et al. [47] identified two common types of multitasking on the web, namely Multiple Tasks and Multiple Session Tasks, and created Multitasking Bar, a simple prototype to support these. The prototype was a browser bar that let users group together webpages into a task, and showed a tab for each task. Jhaveri and Räihä [28] created a prototype tool called Session Highlights to aid cross-session task continuation. Users could drag and drop URLs into URL collections in a workspace, which would create a thumbnail of the webpage, and make it available for later reference. Dubroy et al. [22] did a study of tabbed browsing among Mozilla Firefox users and they found that people prefer using tabs over windows as they cause less clutter and are easier to access. They also found that when people do use windows, they use them for higher-level task grouping and separating work tabs from non-work tabs. Mozilla Firefox now supports Tab Groups, which let users group together similar tabs under a single label and window.

In my system, I wanted to support management of space (windows and tabs) in the web browser in the context of projects. I took an approach similar to Tab Groups, where each project has a single window and any tabs opened in the window belong to the project. The project state is saved by saving the project window's tabs when the project is closed and

state is restored by reopening all the tabs when the project is opened.

2.6 Summary

Information is fragmented into individual information collections of online tools. Information work is frequently interrupted, and difficult to resume after interruption. Researchers have conducted observations, diary studies and interviews to study these problems and in some cases, presented implications for software support based on their findings. Studies have looked at how information workers manage tasks and presented implications for the design of a task management system. Activity-based computing has been explored as a better mapping of real life projects to computing systems and as a better way to manage information. Researchers have also explored the area of window management to allow people to better manage their space and sessions on the desktop and in the web browser. Existing work has tried to come up with solutions to these problems using different approaches, either independently or as a unified solution. Based on the findings and claims in these different areas, I created *Contextinator* to address the problem of information fragmentation in the cloud.

Chapter 3

Design

I designed *Contextinator* based on the findings in the state of the art in this research domain (described in Chapter 2). I identified the following principles to inform the design of my tool:

- Users organize information into projects for organizational purposes [25, 37];
- There is a common group of applications used in most projects (e.g., email, calendar, task management, file manager, etc.) [29];
- Information is fragmented across different collections (or applications) [16, 18];
- Often there is an overlapping set of folder structures, replicated over each information collection (email, files, calendar, etc.) [17];
- Common multiple interruptions require an easy way to capture and restore state, and have a quick way to switch between projects [20, 25, 37];
- A task management system should have the in the way property and should appear in an always visible spot of the working space [12, 14, 25];
- A task management system should support informal priority lists, to ensure completion of near-term priority actions [12];
- Capturing the context while creating a task makes it easier to complete the task later [15, 30];

3.1 Projects: Addressing information fragmentation

A project in *Contextinator* is a collection of browser tabs opened in the same window, series of filters for existing cloud based PIM apps (Gmail, Evernote, Dropbox, etc.), tasks,

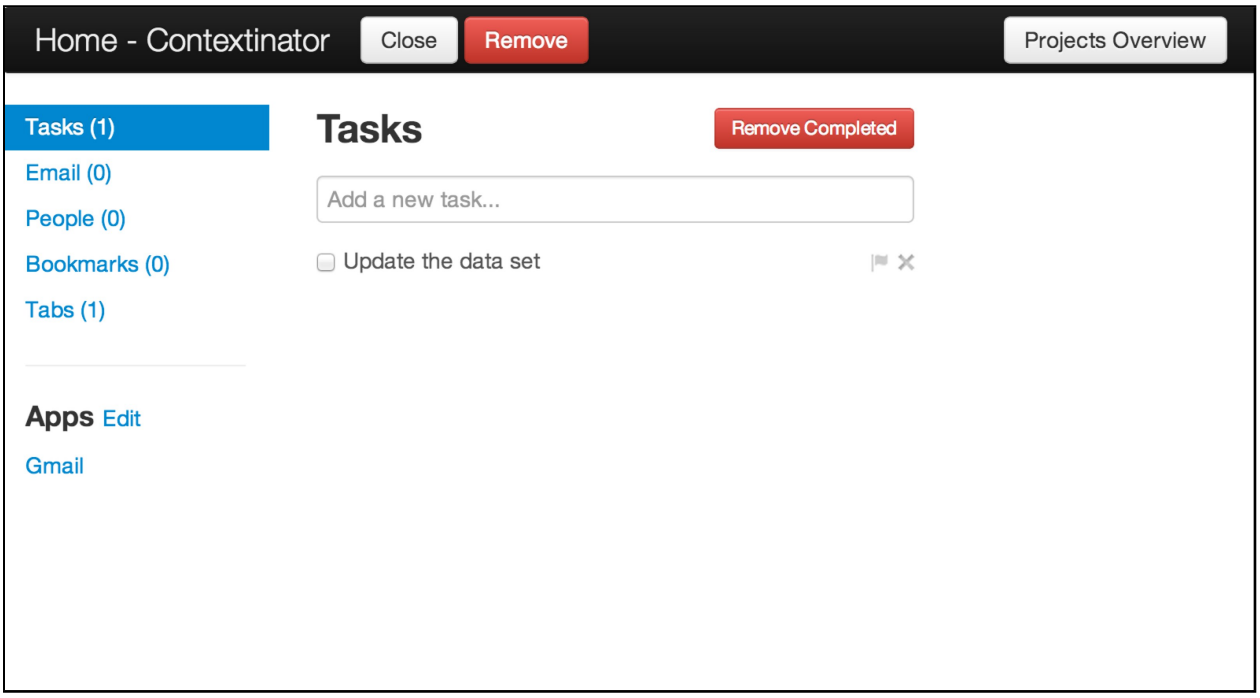

Figure 3.1: Project Homepage.

bookmarks and people. Each project has a homepage (see Figure 3.1) from where all the information associated with the project can be accessed and managed. This page is shown whenever the user opens a project to give the user a sense of where they left off. Users can edit tasks, switch to or close tabs in the project window, open or remove bookmarks, and edit or launch other PIM apps (labeled as Apps in the figure).

3.1.1 Information views

To address the fragmentation of project information across applications, the user can identify an *information view* for a project in each of the supported PIM apps. Most of these apps available on the web have a unique URL that points to a location internally in an online collection. For example, with Gmail, a label has a unique URL, a folder in Dropbox has a unique URL, and the same is true with Evernote and other similar services. These unique URLs allow us to have a direct entry into an otherwise isolated collection.

I support apps used for storing different kinds of information, including email (Gmail), files (Dropbox and Google Drive) and notes (Evernote) [1]. In general, any online service that has a unique URL to point to a sub-part of a personal collection would be appropriate to add to *Contextinator*.

[1] I also support Trello largely because I use it to manage my own projects. This service is not as popular as the other supported apps.

Users can define these *information views* from the *Project Homepage* or while in an app. The *information view* for a project can be a label in Gmail, a folder in Dropbox or Google Docs, or a notebook in Evernote. Defining these *information views* provides a way to coordinate related information fragmented across these different collections under a single project, thus reducing fragmentation.

Shortcut links are provided to these *information views* for the project on its homepage. For Gmail, it fetches the unread email for the project (email that has the Gmail label for the project or email from people belonging to the project) and shows it directly on the homepage. This eliminates the extra step of visiting Gmail to view the unread email for the project.

If users visit the app directly in the web browser while having a project open, they are automatically directed to the project's *information view*. That is, my tool will redirect the user to the particular folder, label, or notebook for the supported apps. This eliminates the navigation to refind a project's information in different applications, each of which may follow slightly different conventions for how the information is organized [16, 17]. Once the user has connected the information in a collection to a project in *Contextinator*, she does not have to remember how to refind information for the project in that collection.

The *information view* is independent of different *organizational units and structures* used in different collections. For example, whereas Gmail uses labels, Evernote uses notebooks, to *Contextinator* both services are just a link away and both stored within the project. This makes my approach flexible and compatible with most modern web-based tools. It also lets users access the full-fledged functionality of these tools, instead of restricting them to a limited interface I might have created.

3.1.2 Project-specific Inbox

Email can play a significant role in task-management within projects and thus I provided a specialized *information view* for email to help users manage their project and work with email. In *Contextinator*, users define an *email information view* by specifying a label and the people in a project. These two together are used to query Gmail and fetch and display if there are new emails from the people in the project or new email with the label specified.

The *Project Homepage* then shows only the unread email related to the project. This lets users avoid distractions by going to their full email inbox. Figure 3.2 shows the unread emails for the "Contextinator" project.

3.2 Projects Overview

The *Projects Overview* in *Contextinator* (see Figure 3.3) is the comprehensive list of all the projects' tasks and emails for the user. Users are able to add or edit tasks in any project.

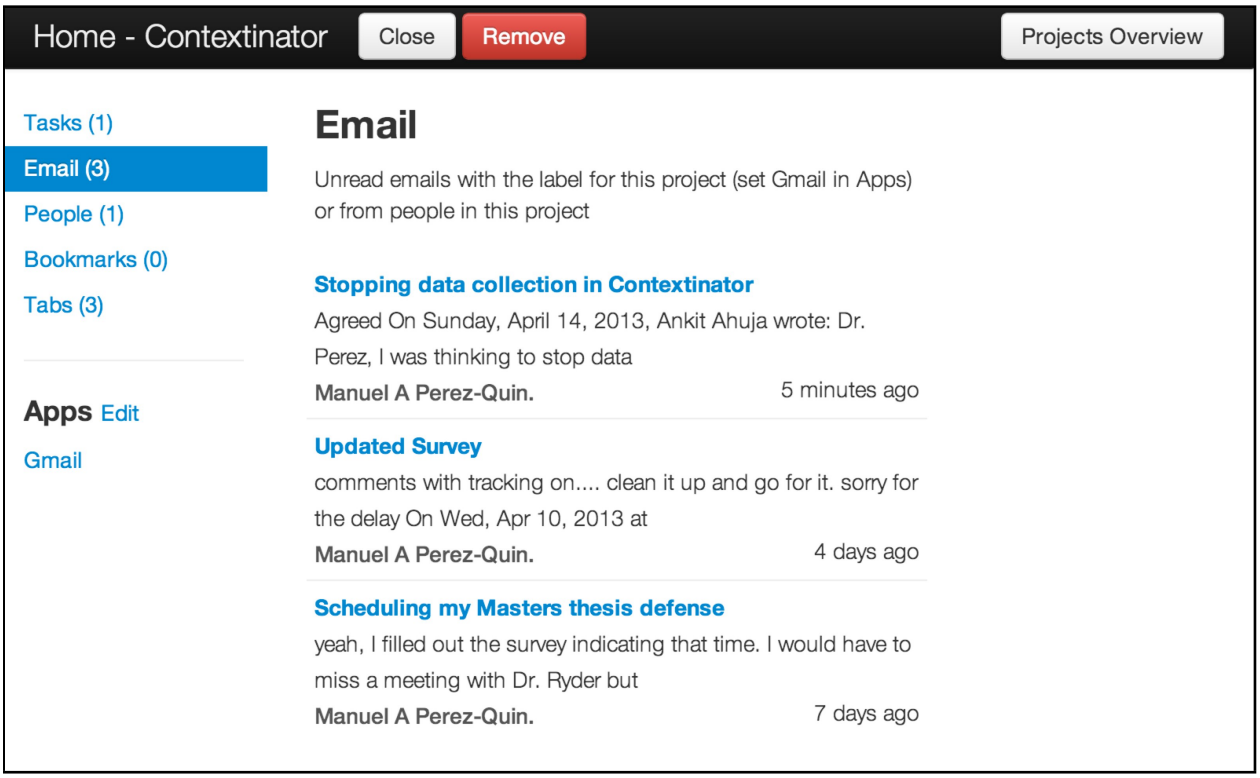

Figure 3.2: Project-specific Inbox showing unread emails on the "Contextinator" project homepage.

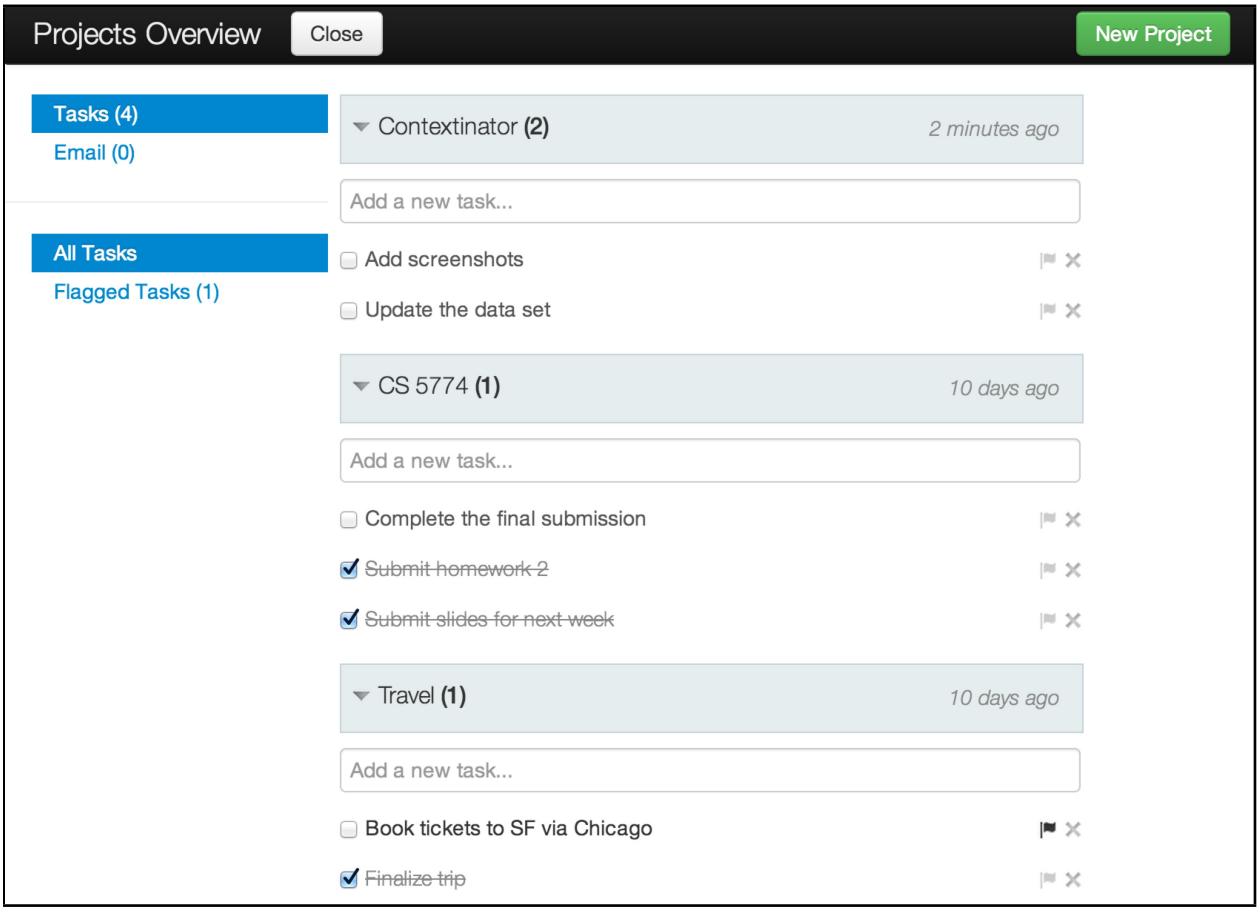

Figure 3.3: Projects Overview showing the tasks across all projects.

Figure 3.4: The *Quick Switcher* showing two projects with the ACM Digital Library homepage in the "Contextinator" project and Gmail open in the "Travel" project.

Users can also filter and see the flagged (prioritized) tasks across all projects. Users can switch to any project from this view. This view also shows the unread emails across all projects, which may indicate tasks for projects.

3.3 Quick Switching between projects

Contextinator lets users associate a single browser window with a project. Any tabs opened in that project window are associated with that project.

To support multitasking, *Contextinator* allows the user to switch between multiple project windows. Users are able to press a shortcut key to see a preview of all their currently open projects (a screenshot of the currently active tab in each project window), and switch between them. This implementation is similar to the Gionarta system [46]. Figure 3.4 shows this view, with two projects, the ACM Digital Library homepage in one and Gmail open in another project.

Because a project is encapsulated in a browser window, users can also manually switch between project windows by using the operating system's default application window switching functionality. Projects are ordered by when they were last accessed by the user. To enable people to resume working on a project, *Contextinator* preserves the state of a project when it is closed by saving the open tabs in the project window at the time of closing the project (i.e. window). The next time the project is opened, all the tabs are restored.

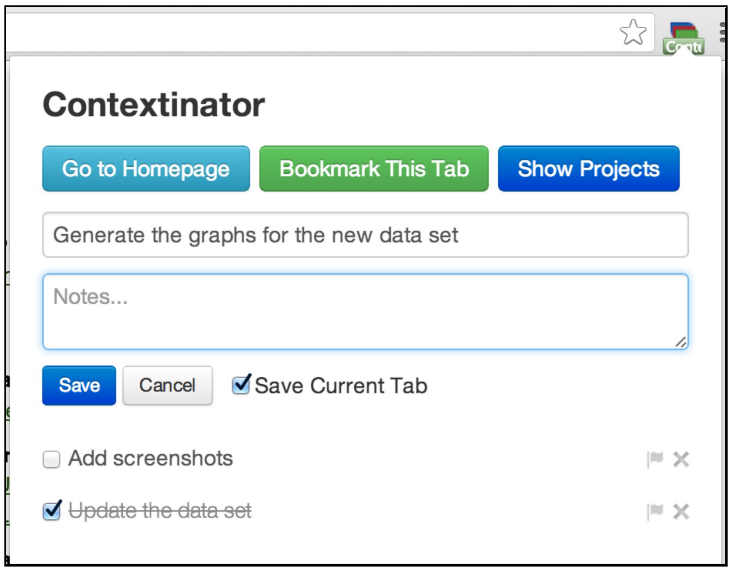

Figure 3.5: Adding a task to the "Contextinator" project.

3.4 Task Based Workflow

Tasks can be seen and edited from various places in *Contextinator*, so that users frequently come across the tasks while they are working. For example, whenever a project is opened, the *Project Homepage* is shown, which by default shows the tasks for the project. This can act as a reminder for the user about the next things to do in that project. I also created a flagging feature to let users prioritize their tasks within and across projects. The project's flagged (prioritized) tasks are sorted to the top on the *Project Homepage*. The *Projects Overview* shows all the projects' tasks, and users are able to filter and only look at the prioritized tasks across all projects.

3.5 Quick Capture

In *Contextinator*, users are able to capture the current context while creating a task. While the user is browsing a webpage, she can create a task, and the current webpage URL will automatically be associated with the task. When she chooses to work on the task next, she can easily go back to the URL. Another way the user can capture the current context is by selecting text on the webpage while creating a task. When the user creates the task, the selected text is attached as notes with the task. Users can also manually associate notes and a webpage URL with a task. Figure 3.5 shows a user saving the current page's URL with a task in a project.

3.6 Summary

I designed *Contextinator* to address several aspects of information work, with focus on reducing the fragmentation of information in the cloud and reducing the adverse impact of frequent work interruptions. The diversity of features in *Contextinator* meant it supported multiple ways of using the tool and people could adopt it in a way that best fit their current working style. Most of the features were based on principles derived from the existing literature. These principles were also the basis of the evaluation of the design of *Contextinator* in Chapter 6.

Chapter 4

Implementation

I implemented *Contextinator* as a browser extension for the Google Chrome browser and it is written using HTML, CSS and JavaScript. To make it easy to modify and instrument for future research experiments, I support modularity by using the *Constructor*, *Prototype* and *Module* design patterns for JavaScript [39]. *Contextinator*'s source code is open source [4] so that anyone can modify and instrument it for future work. At the time of this writing, I have a public webpage [3] describing *Contextinator* and linking to the Chrome Web Store, from where anyone can download and install it.

In this chapter, I describe the data model, architecture, state management and injection of JavaScript into webpages in *Contextinator*.

4.1 Data Model

Contextinator's data model has *Projects* as the main entity and its sub-entities include *People*, *Tasks*, *Apps* and *Tabs*. The Google Chrome Extensions API [6] supports storage and retrieval of data for an extension in a local key-value store. I store projects as an array (where the key is *projects* and value is the array). Each project consists of key-value pairs which represent the properties, state information and sub-entities for the project. The *name* of a project is used as the unique identifier for each project. Figure 4.1 shows how the different data entities are connected.

In addition, I store information for caching and state management:

- *activeProjectIndex* stores the currently active project's index in the *projects* array (for easy retrieval of the active project's information).
- *overviewWindowId* stores the *Projects Overview* window's id.

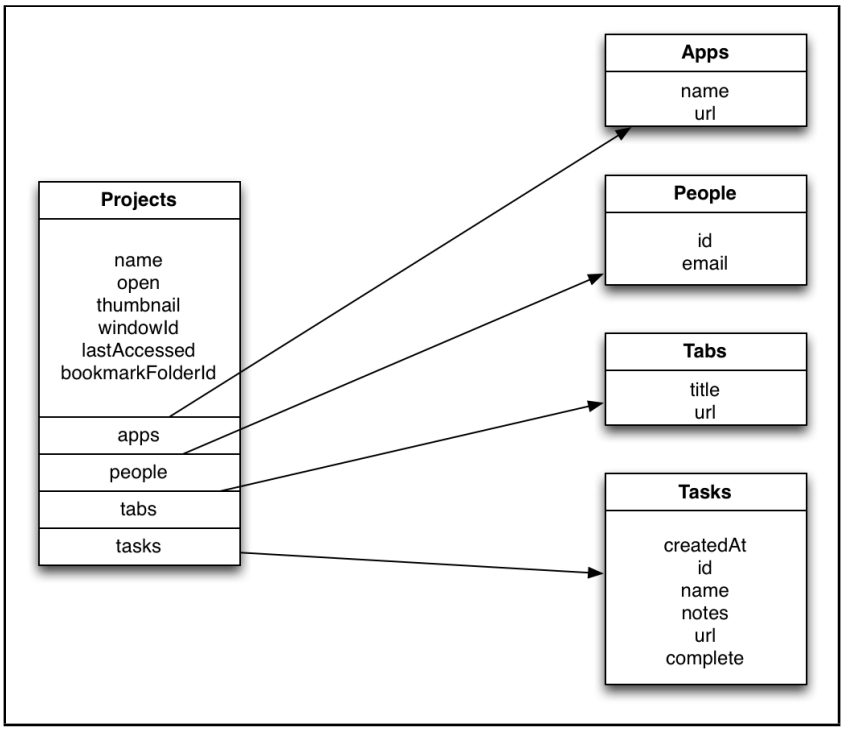

Figure 4.1: *Contextinator*'s data model.

- *temporaryWindows* stores references to the non-project windows.
- *apps* stores information for the supported cloud apps.

  Contextinator can easily support *plugins* (as discussed in Chapter 8) by extending this object.

4.2 Architecture

I divided *Contextinator*'s code into separate modules to represent the different views and data models. Each of the views has a collection of templates, which are declared using a JavaScript templating system [1]. In *Contextinator*, the same information is manipulated in multiple views. For example, tasks are editable on the *Project Homepage*, *Projects Overview* and *Quick Capture*. The parent view in this case includes the tasks module (which consists of a view, a model and templates). Figure 4.2 shows how the different modules are interconnected.

[1] Handlebars.js. http://handlebarsjs.com/

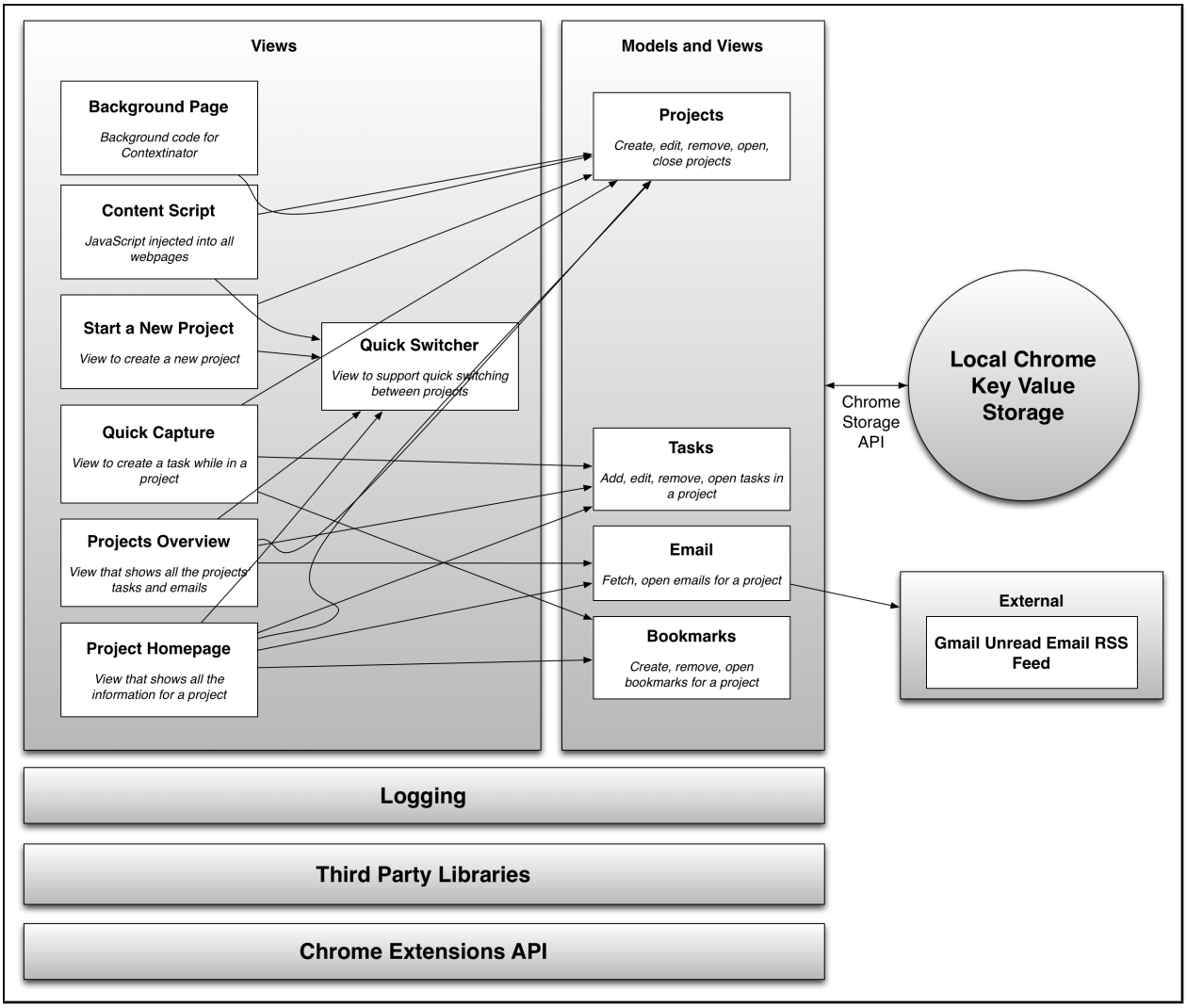

Figure 4.2: *Contextinator*'s implementation architecture.

4.3 State Management

Contextinator keeps track of several events in the browser to keep its data store up to date. For example, it tracks if the user switches focus to a different browser window, so that its data store is updated to the newly activated project.

Chrome lets extensions register for notifications when a specific event occurs in the browser. The *chrome.windows.** API supports adding event listeners for when a window is created, focused or closed. For tracking if the user switched to another window, I defined an event listener for the event *chrome.windows.onFocusChanged*. Whenever this event occurs, my event listener gets called with additional parameters for the event. In this case, it is a single parameter, the ID of the newly focused window. In my event listener code, the newly focused window's ID is compared to the property *windowId* of projects. If one of the project's *windowId* property matches, it means the user activated this project. The *activeProjectIndex* and the project's *lastAccessed* property are updated. The following are some of the browser events tracked in *Contextinator* to maintain the correct state:

- Window Opened (*chrome.windows.onCreated*) - Mark project as open or add non-project window.
- Window Closed (*chrome.windows.onRemoved*) - Mark project as closed or remove a non-project window.
- Window Focused (*chrome.windows.onFocusChanged*) - Mark project or non-project window as active.
- Tab Activated (*chrome.tabs.onActivated*) - Save tabs and update the thumbnail for the active project.
- Tab Removed (*chrome.tabs.onRemoved*) - Save tabs for the active project.
- Webpage Loaded (*chrome.webNavigation.onCompleted*) - Save tabs and update the thumbnail for the active project.

This event listener code is usually executed in the *background page* of the extension, which is an HTML page that runs in the extension process (in the background). It exists for the lifetime of the extension, and only one instance of it at a time is active. These updates to the data store need to be propagated to views elsewhere in the browser, so that they can update their state. For example, if a project window is focused, the *Quick Switcher* needs to update the order in which the projects are listed (projects are listed in the order of most recently activated). A message is broadcast to all the relevant views using the *chrome.tabs.sendMessage* and *chrome.runtime.sendMessage* APIs, which let different views in an extension communicate with each other. In this case, the relevant views will be all

the tabs in the active project's window, since the *Quick Switcher* is displayed in all tabs. The event listener for the event *chrome.runtime.onMessage* in these views then execute the appropriate code to update the *Quick Switcher*. In some cases, such as the *Project Homepage*, I refresh the view when the user activates the tab containing the view. This ensures that the data on the *Project Homepage* is up-to-date without the need to broadcast messages for each of the several events.

4.4 JavaScript Injection

For certain features in *Contextinator*, such as the *Quick Switcher*, additional functionality is added to all the webpages opened by the user in the browser. For example, in this case, an event listener is added for the keyboard shortcut to display a list of the user's projects, as shown in Figure 3.4. Chrome lets extensions inject JavaScript code into webpages in the form of *content scripts*. These *content scripts* are executed in an *isolated world*, which is a special environment where scripts have access to the Document Object Model (DOM) of the page they are injected into but not any JavaScript variables or functions created by the page. In addition to the *Quick Switcher*, I use *content scripts* to redirect the user to the *information views* for a project. When a page from one of the supported apps is starting to load, I check the URL to see if it matches the *information view* URL. If it does not match the URL, I redirect the user to the *information view* URL by setting *window.location.href*. This happens relatively early in the page's loading process, before the DOM is constructed or any other script is run on the page.

Chapter 5

Method

In this chapter, I present the method I used for recruiting participants, data collection (including logging, interviews and an online survey) and data analysis for evaluating *Contextinator*.

I did a field-test evaluation of *Contextinator*, where I deployed *Contextinator* to participants for an average of 18.6 days ($\sigma = 6.1$). I followed that deployment with interviews and a survey to gain further insight into the usage of the tool. This study was approved by the Virginia Tech IRB (#13-008) (see Appendix A for the approval letter).

My evaluation is based on the following procedures:

- First, I deployed *Contextinator* so that participants could use it in their own work setting to see how they would adopt it in their to day-to-day work. The tool logged usage data and I present that data in the next chapter.
- Second, I have rich usage descriptions of individual users actions based on our logging, thus allowing me to report case-study like results for a few chosen users.
- I sent a request to *Contextinator* users to participate in an interview. I present direct quotes and make some observations based on the results of four interviews.
- Finally, I sent an online survey to *Contextinator* users which had questions to assess different parts of the tool.

5.1 Designing the Evaluation

My evaluation design was driven by a set of claims from the existing literature, which were also the basis of the design of *Contextinator*. I started out by grouping together similar

claims from the literature into a list of principles. For each of these principles, I determined the data to collect through logging, survey and interview questions. The following is the list of principles I wanted to test through the evaluation of *Contextinator*:

- Information is fragmented across different collections (or applications) and clustering together information for a project can be useful [16, 18, 32, 37];
- Often there is an overlapping set of folder structures, replicated over each information collection (email, files, calendar, etc.) [17];
- Common multiple interruptions require an easy way to capture and restore state, and have a quick way to switch between projects [20, 25, 37];
- A task management system should have the in the way property and should appear in an always visible spot of the working space [12, 14, 25];
- A task management system should support informal priority lists, to ensure completion of near-term priority actions [12];
- Capturing the context while creating a task makes it easier to complete the task later [15, 30];

Appendix E includes the logging data, interview and survey questions categorised based on these principles.

5.2 Participants

I recruited participants for *Contextinator* using a snowball sampling technique. I emailed the Computer Science graduate student and faculty listservs inviting students and faculty to participate in the study and encouraging them to share the invitation with their colleagues. Students from an undergraduate class were invited to participate and offered extra credit for participating. I announced the tool among colleagues at several high-tech companies including Google and Xerox. Unfortunately, due to corporate policies, only one person from industry could use the tool within their corporate boundaries and let me collect data. I also sent the tool to a few Computer Science graduate students in other universities to distribute the tool in their graduate student listservs. However, I did not get any participants through this channel.

Participants could visit a private website URL to download the tool. On downloading and installing the tool, participants were shown an online consent form which they needed to agree to in order to start using the tool. I also had a public facing webpage [3] that had several videos explaining how to use the different features of the tool.

5.3 Logging

In *Contextinator*, I logged information about users' interactions with the tool. I built the logging mechanism to record most of the user actions, such as creating a new project, switching to a project, closing a project, creating a new task, flagging a task, marking a task as completed, creating a new bookmark, opening email, etc. With each data item logged, I captured a unique user ID, the time stamp of when it happened, as well a some extra information that corresponded to each type of event. For example, I tracked the users' project names and task names, to get a sense of the kinds of projects and tasks the users were creating. Appendix C includes all the user actions collected in the logging mechanism.

This information was collected live and stored locally in the Chrome browser local store. If the user was connected to the Internet, then the data was sent to a central server as it was collected. If the user was offline for an extended period of time, I might miss some data (any actions performed by the user during that time period), since this data was not sent to the central server. During this time, the tool continued to work normally and store the data locally. I used the data stored on the central server for the analysis.

5.4 Interviews

I sent an email to the existing 30 participants after a couple of weeks of use to request volunteers for an interview. I was able to recruit four users for the interview, out of which three (U1, U2, U3) said they considered themselves to be heavy users of the tool and one (U4) that said she did not use the tool as much. U1 was a graduate student and U2, U3 and U4 were undergraduate students. I interviewed all four participants. The interview was a semi-structured interview where I asked broad questions and followed up with specific questions about different parts of the tool. Appendix B includes the questions I asked as part of the interview.

5.5 Survey

I sent an email to the existing 30 participants near the end of the study to invite them to fill an online survey. I received 17 responses. The survey consisted of several Likert style questions to assess how the users used the different parts of the tool and their general work habits. To present the results for the Likert style questions, I have used a combination of weighted average Likert scores on the 5-point scale (1="strongly disagree", 5="strongly agree") and combined percentages of participants that agreed or strongly agreed. The survey questions are included in Appendix D.

5.6 Challenges

The evaluation of *Contextinator* faced several challenges. First, as a tool in the Personal Information Management domain, it is hard to evaluate it using lab-based techniques [34]. Different users make use of tasks lists differently. Different users have different ways of organizing projects. Some users have 1 or 2 projects, while others have 30 (my advisor is an avid user of OmniFocus and has over 30 active projects). Designing and evaluating a tool that can cover such a wide gap is a challenge. I opted to use a field-test approach, similar to that used by Whittaker [48], where he deployed an email program within an organization and collected data that would allow him to describe the use of the tool.

Chapter 6

Results

I collected data for evaluating *Contextinator* through logging, interviews and a survey. This chapter presents the data collected and summarizes the results from the different data sources.

6.1 Participants

The number of participants that completed the consent form and installed the tool was 30. Continuous use of *Contextinator* required a significant commitment from the participants as the tool had to become a part of their day-to-day work. Fifteen users barely used the tool (they created less than two projects and often the project name was "testing" or "something").

For the logged usage data analysis, I used the data from the remaining 15 users. Out of these, eight were active users, that is, users who created more than three projects.

For the survey data analysis, I used the data from the 17 responses I received. I had sent the survey to the 30 participants who had installed the tool.

Table 6.1, Table 6.2 and Table 6.3 show the demographics of the participants that responded to the survey.

6.2 Projects: Information organization

Participants had an average of four projects. Figure 6.1 shows the distribution of the number of projects created by users. There is one user with nine projects, another one with six, three users with five projects, and so on. 70% (12 out of 17) of the participants agreed or strongly

Age	Number of Participants
20	2
21	5
22	4
23	1
24	1
32	1
57	1
Not specified	2

Table 6.1: Self-Reported Age of Participants in Survey.

Gender	Number of Participants
Female	4
Male	13

Table 6.2: Self-Reported Gender of Participants in Survey.

Occupation	Number of Participants
Graduate Student	2
Undergraduate Student	14
Other (i.e. Industry)	1

Table 6.3: Self-Reported Occupation of Participants in Survey.

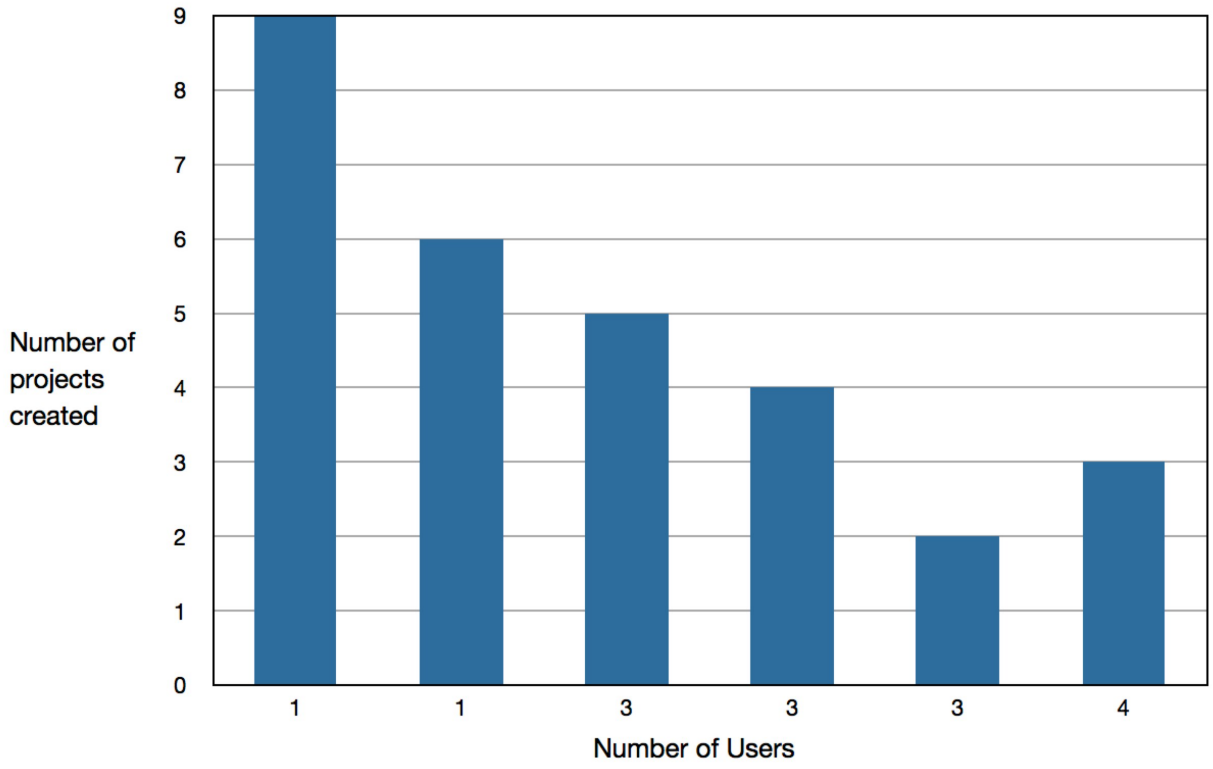

Figure 6.1: Distribution of number of projects created by users in *Contextinator*.

agreed that *Contextinator* was able to handle the number of projects they had (average Likert response = 3.8).

Participants thought about projects very differently, which is reflected in the names of projects created by participants:

- *Class names.* This represented a common type of project, especially since several participants were taking classes in school. Some examples are "Algorithms", "Usability" and "CS 3744".
- *Specific class or research projects.* This was another common category of projects. Examples are "Crypto project", "Leap Study", "3114 Project 3" and "German HW".
- *Non-school related projects.* Some examples are "Gardening", "Bills" and "Tax Returns".
- *Broad categories.* This included a broad range of related activities. Examples are "Web Development", "Shopping", "Courses" and "Things to Check Daily".
- *General categories.* This represented an even broader space for all general activities, which may be unrelated. Examples are "Life" and "General".

In the interviews, participants mentioned multiple ways in which they organized their information into projects. Participant U2 mentioned that he organized his projects into "broad categories":

> *"I have pretty broad categories. I have a General that I just throw stuff in. I have Web Development, so any time I am looking up stuff on stack overflow. I have Shopping, for different stuff I am shopping for...(Participant U2)."*

In contrast, participant U3 thought of projects more as a "list of things" she needed to do. Participant U1 created projects for all his classes:

> *"The first thing I did was make a project for each class I am in. So, when I am in class, I can just open that project and have all the tabs. Esp. for Dr. XX's class, there is like Moodle, Piazza, etc. So that's pretty useful (Participant U1)."*

Three of the four participants I interviewed liked the ability to organize their tabs into projects. Participant U1 defined a project as "A bookmark for an entire window":

> *"I think of them as a bookmark for an entire window. In Chrome if you have a bunch of tabs open, and if I want to come back to all of them at once, bookmark it, make it a project (Participant U1)."*

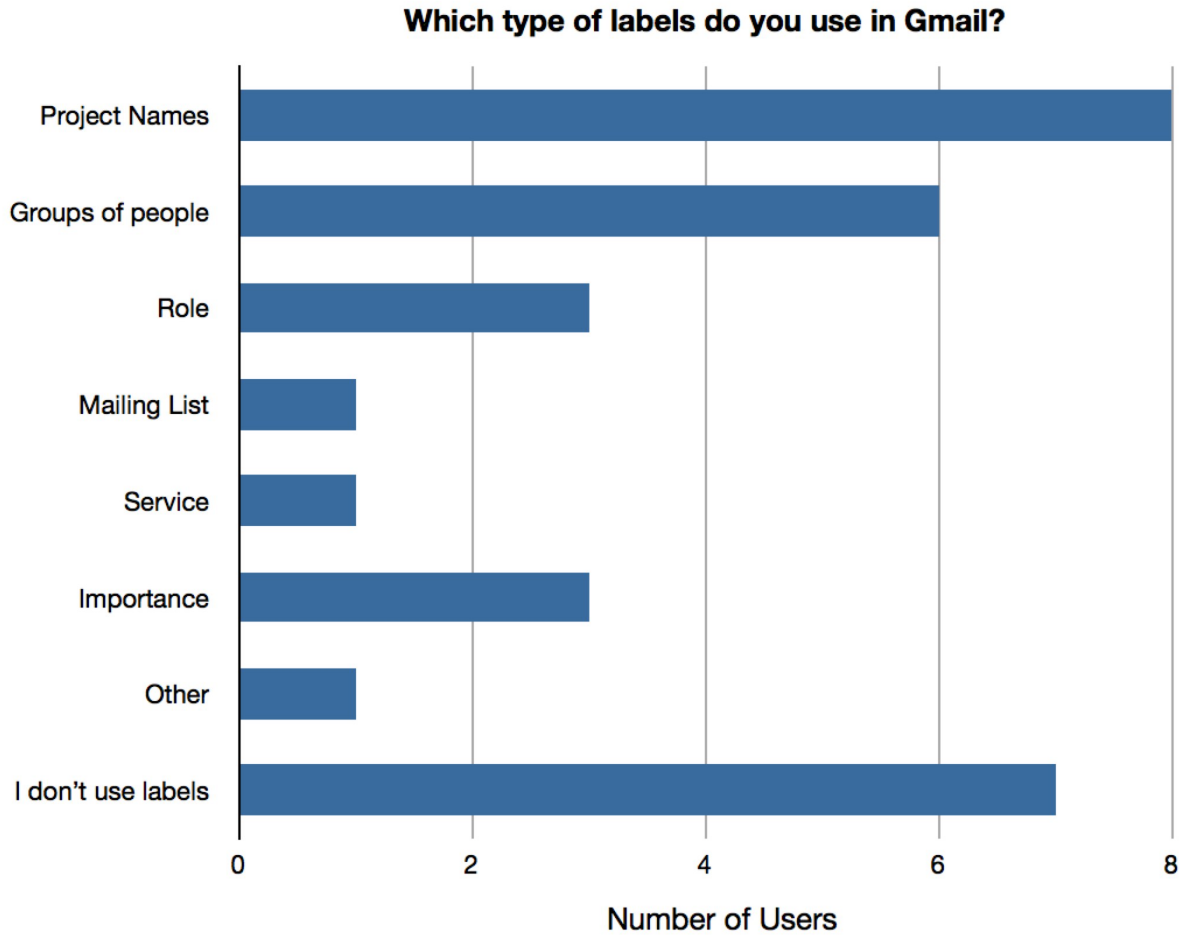

Figure 6.2: Survey responses to types of labels used in Gmail.

6.2.1 Information Collections

In the survey, 59% of the participants agreed or strongly agreed that they use labels, folders, etc. to represent projects in apps (average Likert response = 3.1). For the participants in the survey, a majority of the Gmail labels represent project names, as shown in Figure 6.2. Participants also said they have folders for projects in Google Docs/Dropbox (see Figure 6.3). These results indicate the existence of different information collections for the participants.

Six participants set up apps in *Contextinator*, where each participant usually only set up one or two apps across all projects ($\mu = 2.5$, $\sigma = 1.37$). Gmail, Google Drive and Dropbox were configured by the participants. The app shortcut links on the project homepage were barely used, where only five participants clicked on them twice or thrice.

In the survey, participants rated the usefulness of shortcuts to *information views* on the

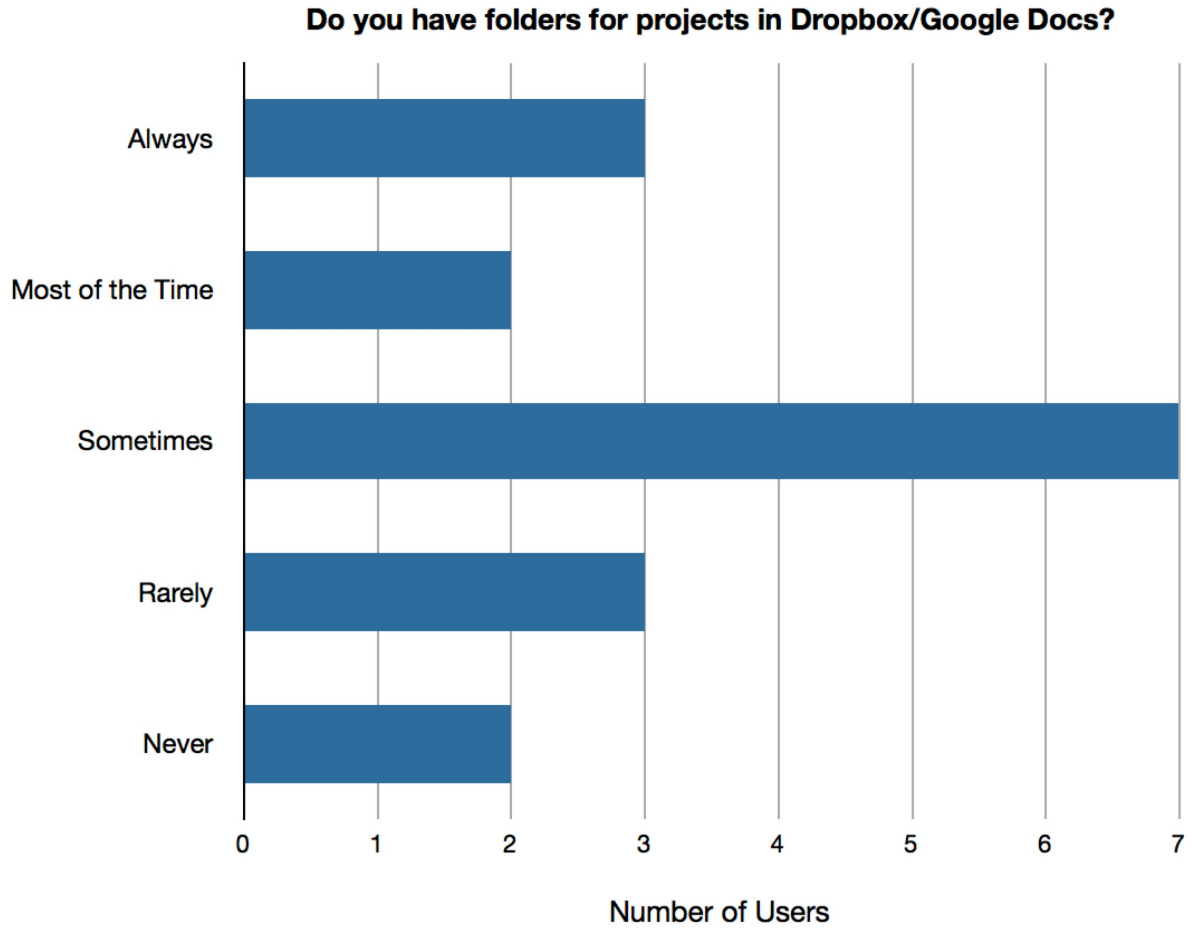

Figure 6.3: Survey responses to folders in Google Docs / Dropbox.

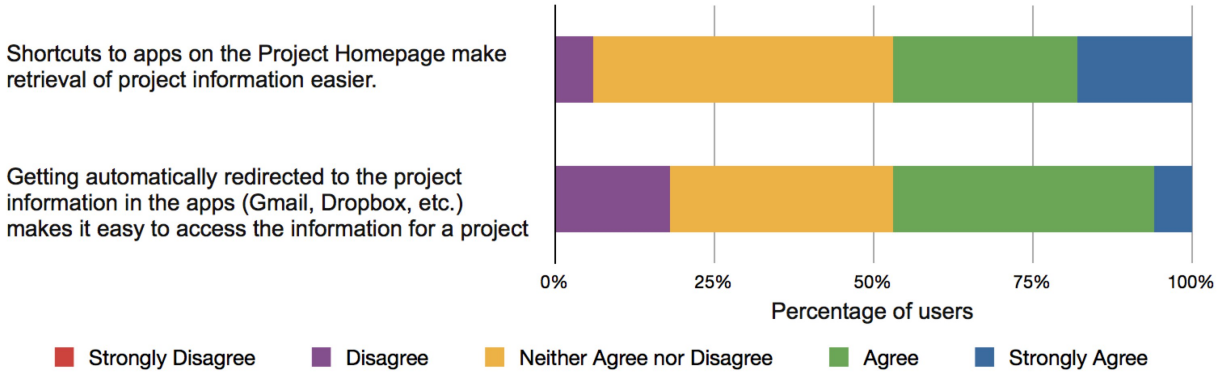

Figure 6.4: Survey responses to *information views*.

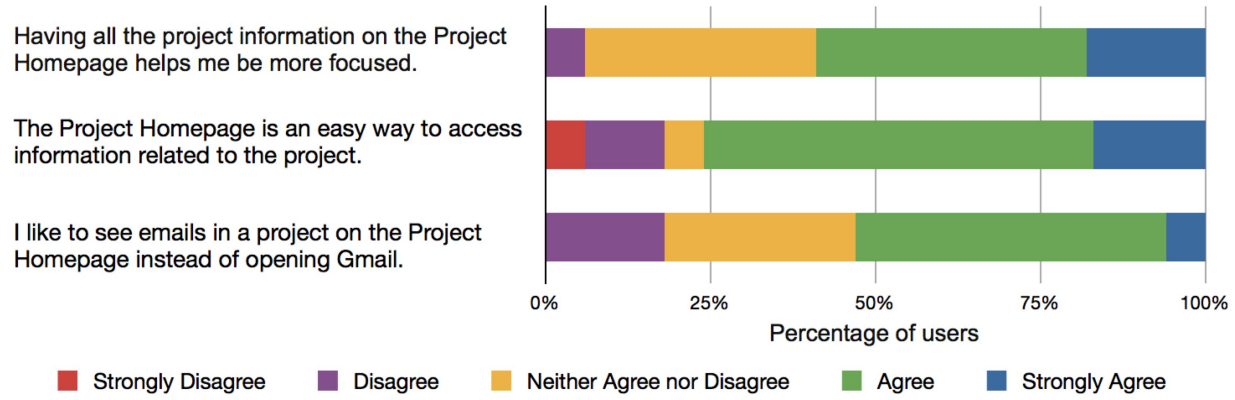

Figure 6.5: Survey responses to *Project Homepage*.

Project Homepage with an average of 3.6, with 47% of the participants agreeing or strongly agreeing that they were useful. The average response to redirection to the project *information views* in apps was 3.4, with 47% of the participants agreeing or strongly agreeing that it made access to project information easier (see Figure 6.4).

Participant U3 found *information views* in *Contextinator* useful:

> *"It is nice on Google Drive...I have so many different folders, its nice to be able to just click it and it goes straight there, rather than having to find it (Participant U3)."*

This is exactly the key use of *information views* I was after. This eliminates much of the information fragmentation as it eliminates the need to navigate to refind the required information within each collection.

6.2.2 Switching between Collection (Apps)

Participant U4 explained how she uses different cloud based apps and has to continually switch between them:

> *"I use Dropbox for my stuff to communicate between my computers. I work for X and I have their folder there. We have an HCI Dropbox. I am bouncing between Email and Dropbox a lot, because there is where a lot of our work happens (Participant U4)."*

To reduce the amount of app switching, I show information for the project on its home-page (including unread emails). In the survey, a majority of the participants (77% agreed or

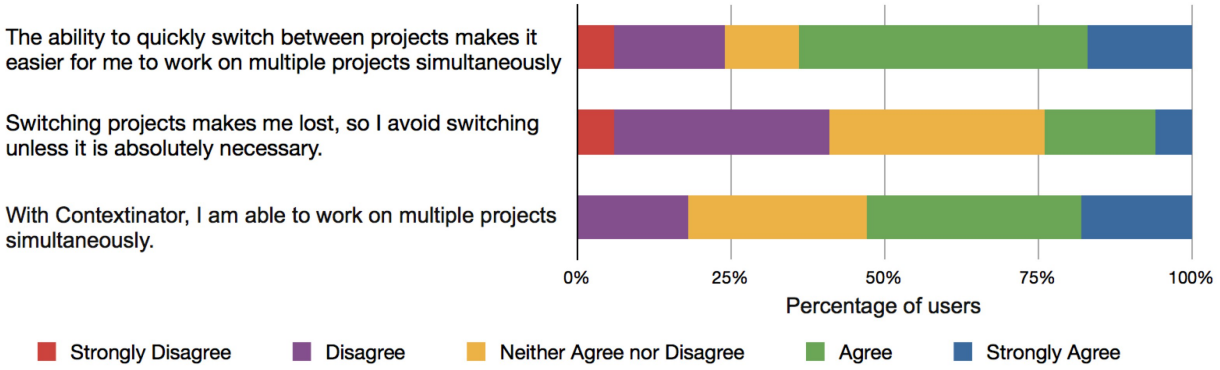

Figure 6.6: Survey responses to switching between projects.

strongly agreed) reported that the *Project Homepage* made it easy to access project information (average Likert response = 3.7). 59% of the participants also agreed or strongly agreed that the *Project Homepage* helped them be more focused (average Likert response = 3.7). These results indicate that participants found it useful to have their project information clustered together (see Figure 6.5).

Two participants (U3 and U4) set up Gmail, but could not use it because it did not show read email for a project:

> *"Usually I get my email so quickly, because of my smartphone, that I don't need that functionality (people). Because it (project homepage) doesn't keep the emails there, it doesn't do much for me (Participant U3)."*

The people feature was hardly used in *Contextinator*. Only three participants added people to their projects and only one of them clicked on an email link (which uses people for filtering).

6.2.3 Project Switching

Contextinator users made an average of 14 switches between project and / or non-project windows per day (on days the system was used), ranging from the maximum of one user switching an average of 44.7 times daily over a period 21 days, to the minimum of a user switching an average of 2.3 times daily over a period of 11 days ($\sigma = 12.3$).

A majority of the participants (65% agreed or strongly agreed) found that it was easier to work on multiple projects simultaneously with the support for quick switching between projects (average Likert response = 3.5). 53% of the participants also agreed or strongly agreed that they were able to work on multiple projects in *Contextinator* (average Likert response = 3.5). These results show that participants found the support to switch between

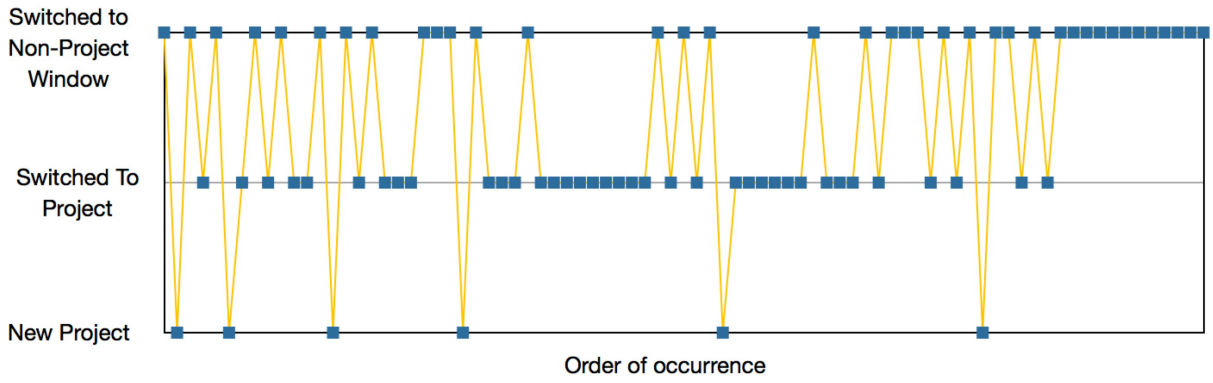

Figure 6.7: Switching between project and non-project windows in *Contextinator*.

projects useful when working on multiple projects (see Figure 6.6). When asked if participants avoided switching between projects because it made them lost, the average response was 2.8, with 24% of the participants agreeing or strongly agreeing.

Three of the participants mentioned in the interview that they did not use the tool to decide which project they wanted to work on but the tool made it easier to work on multiple projects at once.

A couple of participants described how they often transitioned to another project. They did not realize they wanted to switch to a project until they had a few tabs open related to it. This happened usually when they remembered they needed to do some work in another project (self-interruption):

> *"A lot of the times I would just open a bunch of new tabs, and not necessarily look for an existing project first...I start googling something, and I have five tabs open. And then I realize, actually this should really go into the VTS project...(Participant U1)."*

Figure 6.7 shows how participant U1 frequently switched back and forth between a project and a non-project window. This indicates the requirement of flexibility in an activity-based system to support non-project spaces, to hold applications and information not belonging to any project [32]. A couple of participants had generic non-work related projects such as "Life" and "General", which they would often switch back and forth to.

6.3 Doing work

Thirteen participants used the task feature (created at least one task). They created 6.8 tasks on an average (σ = 13.4) and completed an average of 4.7 tasks (σ = 10.9). Participants

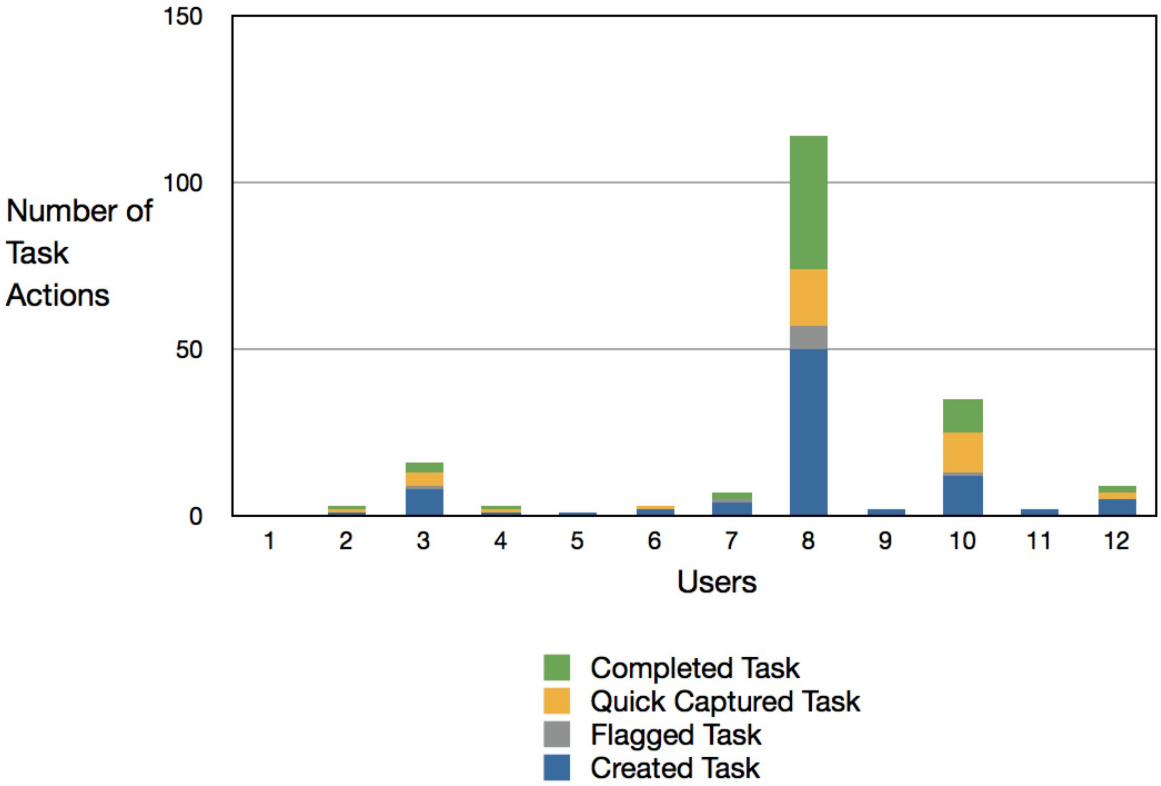

Figure 6.8: Number of tasks created, flagged, quick captured and completed by users in *Contextinator*.

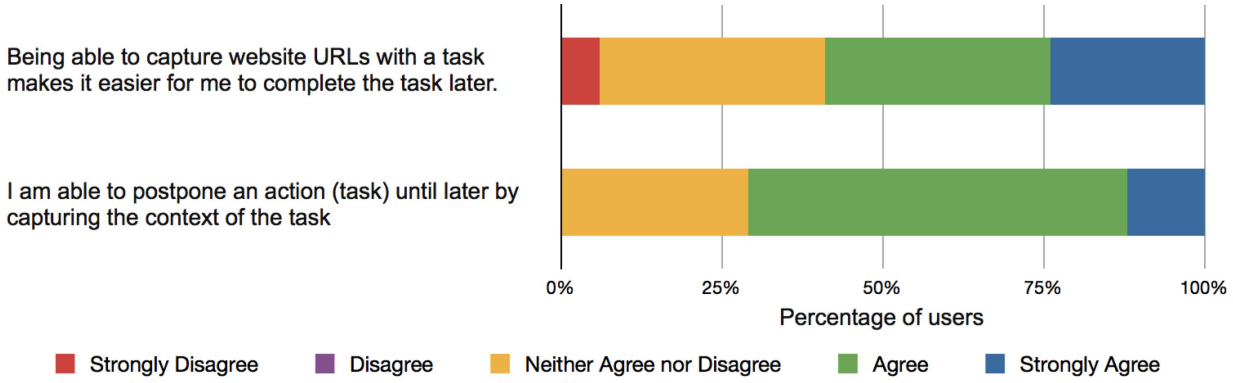

Figure 6.9: Survey responses to Quick Capture.

rarely used flagging. On an average, each participant flagged 0.8 tasks ($\sigma = 1.9$). Figure 6.8 shows the number of tasks created, quick captured, flagged and completed by users.

Contextinator makes tasks visible at multiple places (*Project Homepage*, *Quick Capture*, *Projects Overview*). However, this still requires a user action to view the tasks, as was indicated by participant U4. She wanted tasks to be always visible in her workspace to match her existing work style:

> *"I would copy and paste all the tasks that I need in my Word document and look at it that way because it is easier to work on my homework...If I could view the tasks on a sidebar with whatever I am doing in the web browser, I don't have to click to the other tab and then click to the other thing...(Participant U4)."*

This indicates the need to further increase task visibility in the user's workspace [14, 12, 25].

6.3.1 Quick Capture

Participants quick captured (added a URL or note to a task) an average of 3.1 tasks ($\sigma = 5.2$). Participants reported in the survey that capturing website URLs made it easier for them to accomplish the task later, with an average response of 3.7 and 59% of the participants agreeing or strongly agreeing. None of the participants disagreed that capturing the context of a task enabled them to postpone a task for later (average Likert response = 3.8). These results indicate that participants found quick capture useful (see Figure 6.9). Figure 6.10 shows the types of URLs participants reported they stored with tasks.

Participant U3 was by far the most active user of the tasks feature (See Figure 6.8). She found the ability to capture context in the form of a URL or notes with a task useful:

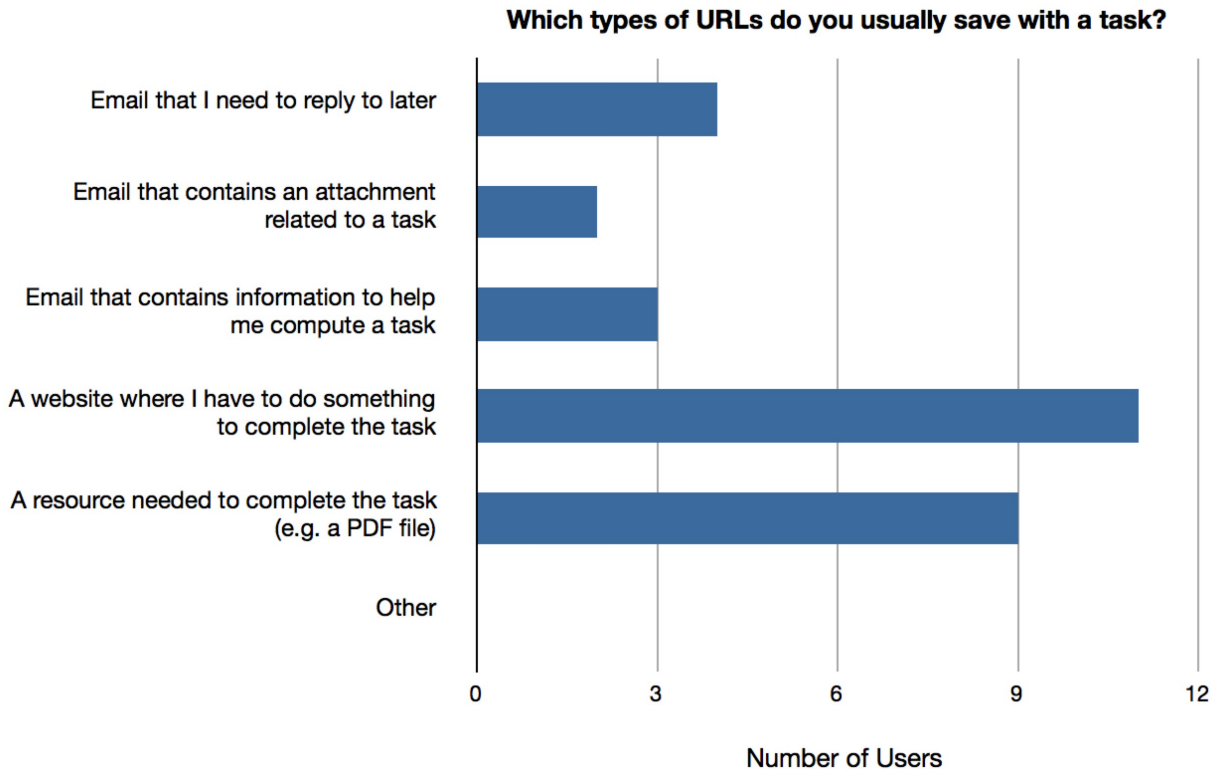

Figure 6.10: Survey responses to types of URLs quick captured.

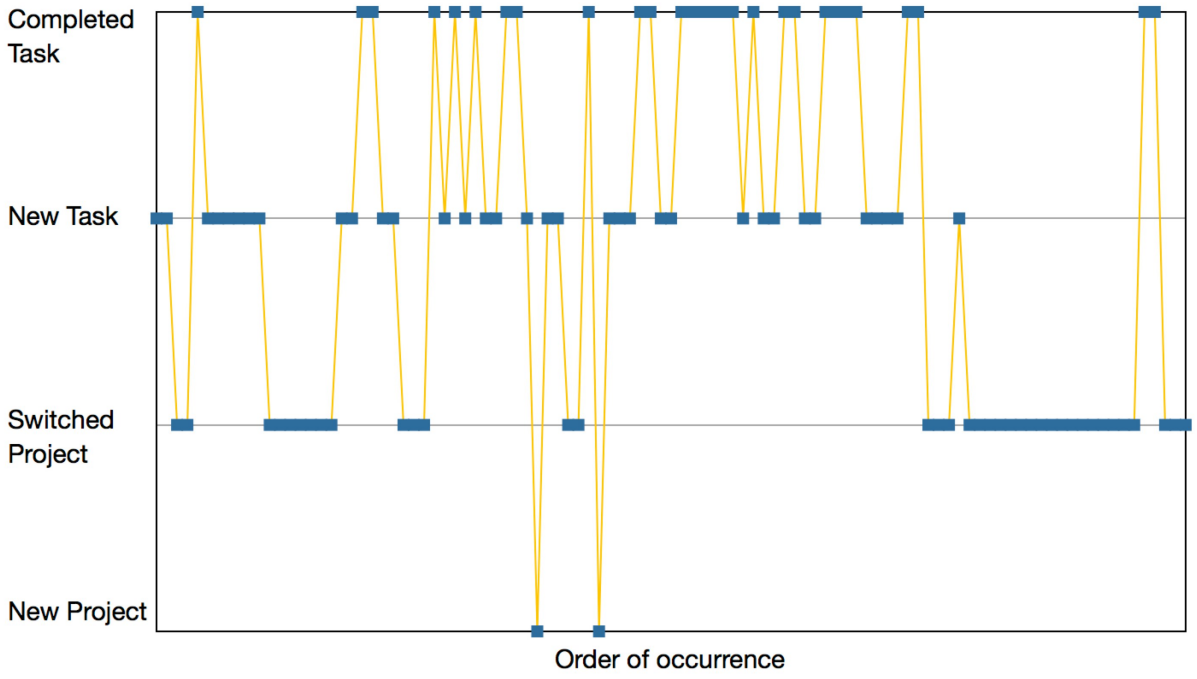

Figure 6.11: Preparatory approach to tasks and projects in *Contextinator*.

> *"...you can get very easily distracted because when you know what your task is, then you have to find everything you need for that task. So, if you could put everything that has to do with that task, it cuts out so much transition time...you can just immediately work on that task (Participant U3)."*

Figure 6.11 shows one type of use of projects and tasks that I observed. It depicts a *preparatory* approach [48], where the user creates tasks ahead of time, and then completes them over a longer period of time. The user created a new project, followed by several new tasks, and then marked them as completed later.

Contrast that with Figure 6.12, where the user is taking more of an *opportunistic* approach. She creates new tasks as and when required and marks them as completed in the near future. Here, the user does not plan things she needs to do in a project ahead of time.

6.4 More Observations

6.4.1 Wrong Fit for Participants

Since I had only one participant from industry, I do not have data to reflect on their use cases. Through the logging data and interviews, I found that students had a limited view of

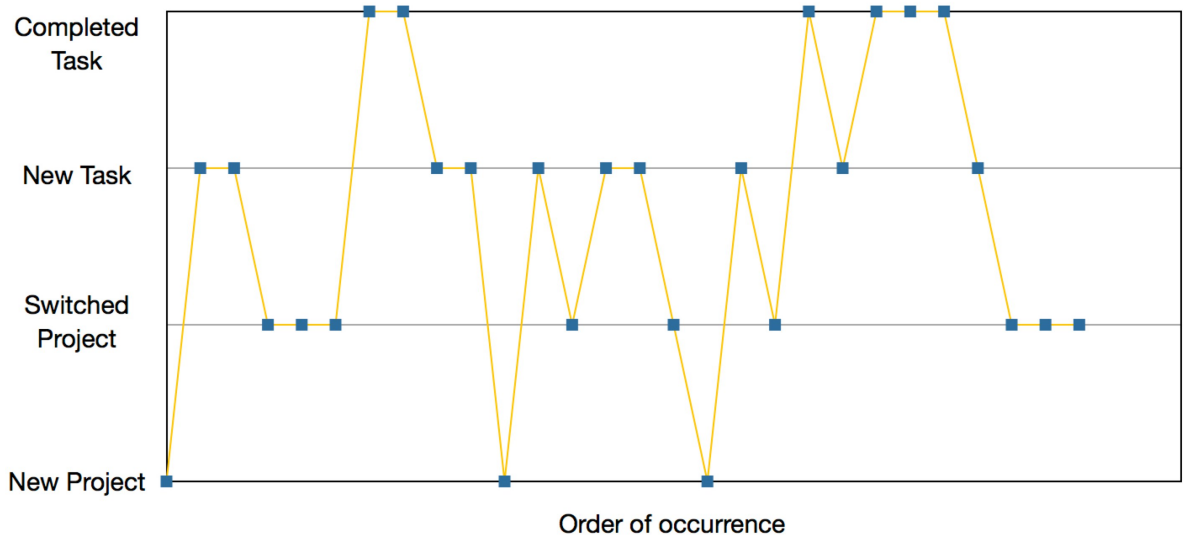

Figure 6.12: Opportunistic approach to tasks and projects in *Contextinator*.

projects in some cases (e.g. classes, projects for classes):

> *"When I was first using it, the title project...it made me feel like it should be like a school project or a research project...My first impression of looking at it was that it was like a school thing or a research thing. It definitely threw me initially (Participant U4)."*

Often, students had few projects or did not have information that they found overwhelming to manage. For example, participant U4 only had two projects.

Participant U1 used another task management system and did not use it to decide what to work on next (which was opposite to how the task management system was designed in *Contextinator*):

> *"I don't normally go there (task manager) to decide what to do. I go there so that I don't forget things to do...When I decide it is time to work on a project, it is time to work on a project because I have this particular thing to do. So I come in already knowing the task that needs to be done...(Participant U1)."*

6.4.2 Positive experiences

In the survey, a majority of the participants (76% agreed or strongly agreed) reported that reopening tabs for a project helped them resume working on a project, with an average

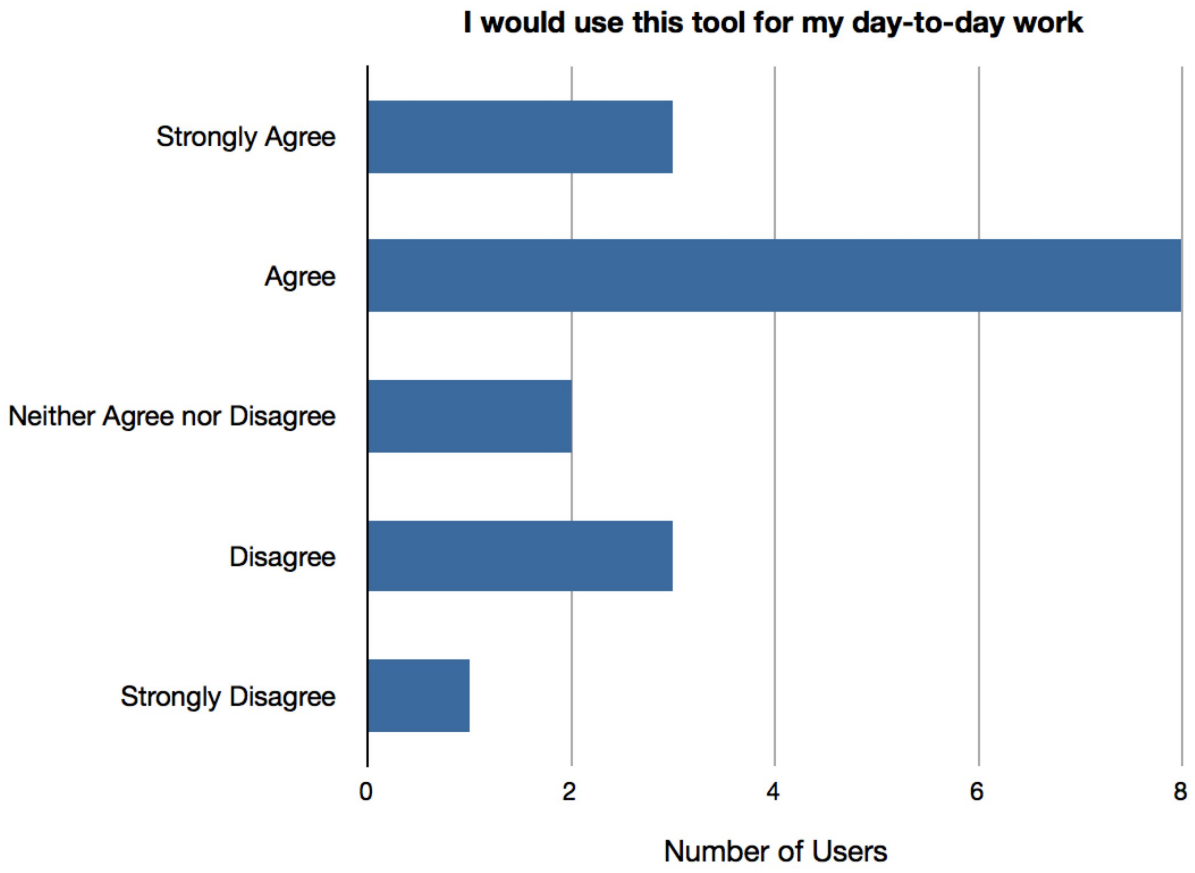

Figure 6.13: Survey responses to "I would this tool for my day-to-day work"

.

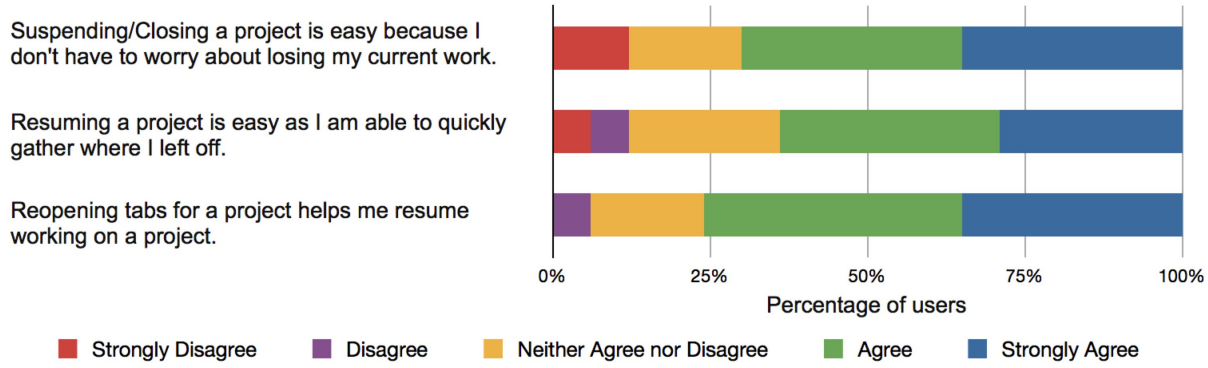

Figure 6.14: Survey responses to project state.

response of 4.1. Participants (70% agreed or strongly agreed) also reported that they could suspend or close a project without the anxiety of losing their work, with an average response of 3.8. These results show that saving the project state helped the participants resume working on a project later and they did not have to worry about losing their work (see Figure 6.14). Participant U4 also mentioned this in the interview:

> *"The biggest plus is being able to have the tabs saved and I don't have to look at them or not close my Internet to keep all those tabs (Participant U4)."*

One of the participants mentioned in the survey that *Contextinator* helped them be more organized:

> *"I liked how I could work on multiple projects and it allowed me to stay organized."*

Several participants found organizing tabs into projects useful. Participant U1 said:

> *"I guess since I started using this tool, each project is in its own window, which makes things easier. It used to be I just open lots of tabs and then you have so many tabs open you don't know what's going on. And then you just have to close a bunch of tabs. Then you don't know where you were before (Participant U1)."*

Another participant mentioned that *Contextinator* helped them be more focused while working:

> *"I liked how easy it was to stay focused on an individual project. I so often get distracted by other things going on, and contextinator really helped with that."*

Participants liked the ability to quickly switch between projects (see Figure 6.6). Participant U3 found it useful to set up the *information view* for Google Drive. Participants U2 and U3 found the task feature useful to define intermediate tasks for their projects.

Overall, 65% (11 out of 17) of the survey participants agreed or strongly agreed that they would use this tool for their day-to-day work (see Figure 6.13).

6.4.3 Limitations

Three out of the four participants I interviewed wanted the ability to set deadlines for tasks. Participant U4 used two computers (desktop and a laptop) interchangeably, and wanted to be able to link them and have the same projects at both places:

> *"From my personal experience in using it, it made it a little bit less convenient...I use Dropbox between the two computers and I set up users on my browser so I share the same bookmarks and everything...(Participant U4)."*

Two participants wanted to be able to share their projects with groups of people, and accomplish tasks in a project together:

> *"If you'll be able to share different projects with people...that would have been really useful in my X class where we are all doing the same project and we could have similar bookmarks...(Participant U4)."*

A participant mentioned in the survey that they wanted the ability to combine multiple projects into a single window:

> *"Usually I work on one project at a time. But I have found it inefficient to work on two projects at once because they use different windows. Possibility of making some option to temporarily combine projects in a window? and somehow separate again after?"*

Another participant also wanted the same capability, they wanted projects to open as new tabs instead of new windows. A couple of participants mentioned (U3 and U4) that they wanted to able to see read email for a project on the homepage.

Chapter 7

Discussion

I implemented several features in *Contextinator* based on claims in the existing literature, and some of my findings match those claims. These can also act as the requirements for similar tools in the future. In this chapter, I summarize the findings from the evaluation of *Contextinator*.

7.1 Information views

I set out to solve the problem of information fragmentation by introducing the concept of *information views* in *Contextinator*. Users could identify an *information view* for a project in each of the supported PIM apps by specifying a unique URL that pointed to a location internally in the apps.

- Few participants set up and used *information views*. Participants mentioned in interviews that they found it difficult to set up *information views*, which could be a possible reason for this lack of use.
- The participants who did use *information views* had very positive opinions about it (see Figure 6.4). Participant U3 mentioned that direct navigation to a project's folder in Google Docs was useful (where she had several folders). This was the key use of *information views* I was after.
- Several participants suggested other ideas to extend the concept of *information views*. Participants U3 and U4 set up *information views* for Gmail, and said that they would have found it useful if it made all the project's emails visible on the *Project Homepage* (instead of only unread email). Participants also wanted support for other apps they used.

The participants who set up *information views* and used them had a strong liking for the idea. However, the implementation of *information views* could be made better. Some participants found it difficult to grasp the idea of an *information view* and their functionality. Conveying the idea of an *information view* to the users in simpler terms would be useful. Some of the participants also found the steps to set up an *information view* confusing. Even though users could define the current view while using an app as an *information view*, few users noticed it. Further reducing the friction in setting up an *information view* will improve its usability. If the ability to authenticate the user with the apps is implemented (discussed later in Chapter 8), displaying a list of all the *organizational units* in an app on the *Project Homepage*, and letting the user select the one for the project would be useful. This would make the process of setting up an *information view* more streamlined than copying and pasting URLs.

With better usability, extensibility to support any web-based tool and stronger integration into *Contextinator*, *information views* can bring back the context lost amid the fragmentation of information across different web based tools.

7.2 Clustering together Project information

In *Contextinator*, I displayed all the information for a project on its homepage. Participants indicated a strong liking for this idea, and thought this helped them be more focused and enabled easier access to information (see Figure 6.5). Extending the concept of *information views* to display snippets of information from web-based tools on the homepage will create further aggregation of information on the *Project Homepage*. This matches the claims made in the existing literature that clustering of project information together is useful [37, 32].

7.3 Switching between Projects

In *Contextinator*, users switched between projects frequently and found it to be useful (see Figure 6.6). This is similar to findings reported in the literature by González and Mark [25, 37].

Switching between projects was easy but users did not always know when to switch between projects. A typical user behavior was that the user started to work on a new project while still within the old project and later realized she should switch to a different project to save her progress. This eventually resulted in the user individually dragging and dropping tabs from one window to another. This was chaotic since the project windows were by default occluded behind the current project's window, and the user needed to have both windows visible at the same time to accomplish this. Some users also wanted to be able to work on multiple projects in the same window. This indicates the need to support an easier and flexible transition between projects. New functionality is required to help users move some

of their work (tabs) from one project (window) to another.

Contextinator saves the tabs for a project when it is closed. When work on the project is later resumed, the tabs for the project are restored. Participants found this helped them resume working on a project (see Figure 6.14). This is similar to findings in existing literature which claim that it is important to provide cues to help users resume working on a project [20, 25, 37].

7.4 Identifying a Project Window

Participants mentioned in the survey they got lost when they switched projects (see Figure 6.6). Participant U3 avoided switching projects due to this reason. Participants were often not sure which project they were in since the current window was not prominently labeled with the project name. This was a limitation imposed by my implementation architecture. It is imperative for a system like *Contextinator* to have a fixed visual marker identifying the current workspace, similar to the current application name displayed in the menu bar on OS X.

7.5 Visibility of Tasks, Quick Capture and Flagging

I created the task management system in *Contextinator* to let users capture and manage tasks for their projects to make accomplishing tasks easier.

- Few participants actively used tasks in *Contextinator* (see Figure 6.8). Some participants said that they do not have many things to do, so they do not need a task management system. The tasks feature was a wrong fit for these participants.
- One of the participants (Participant U4) mentioned that she wanted the tasks to be always visible, instead of having to switch to different views. Even though I made tasks visible at several places, it still required a user action to view the tasks (the tasks were not always visible). This indicates the need to keep tasks always visible in periphery of the workspace while working on a project to serve as a reminder to the user and be in the way. This matches the findings in the existing literature [14, 12, 25]. Participant U4 suggested to display the tasks in a sidebar in *Contextinator*, so that they are always visible while browsing any tab in a project's window.
- Participants found the ability to capture context with a task useful (see Figure 6.9). Implementing ways to make capturing and returning to a task more seamless and quicker would be useful (e.g. keyboard shortcut to capture current webpage as a task, instead of multiple steps involved in adding a task).

- Flagging for tasks was barely used by the participants. The data is inconclusive if flagging is useful or not needed. Most of the participants did not use tasks actively and this feature would have been useful only if the participants had a lots of tasks and needed to prioritize certain tasks.

7.6 Lack of use of Projects Overview

I created the *Projects Overview* to let users review the tasks they needed to do across projects. However, this view was barely used by the participants. Part of the reason could be that participants did not have numerous projects that they were working on, so they might not have needed this support. Several users also did not use tasks and so this view had limited use for them. Participant U3, who used tasks actively, found it useful to look at the information for all her projects together. Since this view was not actively used, I do not have enough data points to determine if it is useful or not needed.

7.7 Summary

Participants actively adopted *Contextinator* in their daily workflows and created a wide variety of projects. *Contextinator* required a significant commitment from the participants as it had to become an integral part of their day-to-day work. Several participants did not have information or projects they found overwhelming to manage, whereas most features in *Contextinator* were designed for users with a lot more projects. Despite this, they actively used *Contextinator*, even for a few projects. 65% (11 out of 17) of the participants either strongly agreed or agreed that they will use *Contextinator* for their daily work. Participants naturally divided their work activity into projects and a subset of the features in *Contextinator* were useful enough for the participants to keep using it. These subsets of features varied for participants. Whereas some participants used *Contextinator* primarily for saving tabs, others used it as a task management system integrated into the web browser. This flexibility was one of the overall design goals of *Contextinator*, so that it is useful for a broad variety of users and not just information workers. Such a flexible approach is important if we want to move people away from the desktop to the activity-based paradigm, instead of building yet another specialized tool for information workers.

Contextinator has shown potential in its approach to solve the problem of information fragmentation in the cloud. Its deployment and evaluation have highlighted some of the main problems that need to be addressed and solutions that can help reduce the fragmentation of information across web-based tools. It has also highlighted requirements for software support to help with information work. *Contextinator* is a good launchpad for future exploration in this research area to build tools that address the fragmentation of information and information work amid the growing popularity of cloud based applications.

Chapter 8

Conclusion and Future Work

People have vastly different concepts as to what a project is and designs of similar tools should handle this. We need to keep in mind that users make use of more than just one tool. With the explosion of Personal Information Tools in the commercial space to manage different kinds of information, information workers pick the tools that best suit their workflow. They may even switch frequently between tools to fit their changing organization schemes and work styles. A system like *Contextinator*, that seeks to solve the fragmentation of information across different tools, must be extensible and should work with these existing tools. I feel that *Contextinator* has the potential to create recreate the context lost amid information fragmentation in today's web-based tools.

8.1 A broader approach to *information views*

In *Contextinator*, the user can identify an *information view* for a project in each of the supported PIM apps by specifying a unique URL that points to a location internally in the app. This concept can be extended to support any web-based service by using a plugin-based approach, where a plugin can be defined for any web-based service in *Contextinator*.

A plugin for a service will be a simple semi-structured data file (JSON), similar to what I am using in the current implementation of *Contextinator*. The following is an example for the Gmail service, where *base_uri* represents the string which if present in the URL, the user will be redirected to the project's *information view* and *ignore* represents a set of URLs for which the user should not be redirected (e.g. login page for the service).

```
{
  name: "Gmail",
  base_uri: "https://mail.google.com/mail/u/",
  ignore: [
    "https://accounts.google.com/ServiceLogin"
```

```
  ]
}
```

In the interviews, I found that participants wanted a deeper level of integration with their cloud based apps in *Contextinator*. A couple of participants indicated they would have found viewing email on the *Project Homepage* useful if it was all of the project's email and not just the unread email.

One example of such an integration with apps would be to show snippets of information on the *Project Homepage* from apps (similar to email). This would reduce the amount of app switching users need to do and further aggregate project information on the *Project Homepage*, which was appreciated by participants (see Figure 6.5).

Contextinator can support such an integration with any web-based service that supports REST based APIs for:

- Authenticating the user

  Contextinator will authenticate the user with the web-based service so that it can access its data. Web-based services commonly support this through protocols such as OAuth [9].

- Getting a list of *organizational units* (List of folders, labels, etc.)

  Most web-based services that support organization of information have a way to access the basic *organizational units* such as folders, labels, etc. This is because data for these services is device agnostic, and needs to be accessed, displayed and modified on different platforms and devices. For example, the Dropbox API supports getting a list of all folders for a user [5].

- Getting the data for an *organizational unit* (List of files in a folder, etc.)

  Web services also have APIs to access data inside *organizational units*. For example, Dropbox supports getting a list of files in a folder for a user [5].

Contextinator can connect to these services, authenticate the user and get a list of their *organizational units*. The user will be able to pick one of those as the *information view* for a project. *Contextinator* can then access information for this *information view* for the project and display it on the project's homepage.

The user can define project-specific templates to define how to display the fetched information for an *information view*. For example, whereas a user may want to display a list of upcoming deadlines from her Calendar for a project that represents a class she is taking, she may want to have quick access to files stored in a Google Docs folder for a project that represents writing a paper for a conference. These views could also provide additional functionality than simply displaying information. For example, there could be a plugin for Dropbox that displays files

from the project's Dropbox folder on the *Project Homepage* and lets users open, edit and drag and drop files in this space. This would then act as a piling area for the users, similar to the desktop in Gionarta [45]. This would enable users to create a personalized workspace for each project, similar to how information workers create their physical workspaces [36]. Each of these services will have their own API structure and response format, so in order to support these different services, the plugin for a service would define the API endpoints and templates to define the functionality of the plugin's view on the *Project Homepage*.

Contextinator can support a set of default plugins, and power users can create their own which can then be reused by other users. Such a collaborative workflow can be achieved by creating a community of *Contextinator* users. An example of such a community is for the tool Stylebot, where users are able share and reuse custom style sheets for popular websites [10].

Bibliography

[1] attachments.me. https://attachments.me.

[2] Cloudmagic. https://cloudmagic.com/.

[3] Contextinator. http://contextinator.cs.vt.edu.

[4] Contextinator github code repository. http://github.com/ankit/contextinator.

[5] Dropbox rest api. https://www.dropbox.com/developers/core/api#metadata.

[6] Google chrome extensions api. http://developer.chrome.com/extensions/api_index.html.

[7] Mailbox. http://www.mailboxapp.com/.

[8] Mailpilot. http://www.mailpilot.co/.

[9] Oauth. http://oauth.net/.

[10] Stylebot. http://www.stylebot.me/.

[11] A. D. Balakrishnan, T. Matthews, and T. P. Moran. Fitting an activity-centric system into an ecology of workplace tools. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 787–790, New York, NY, USA, 2010. ACM.

[12] V. Bellotti, B. Dalal, N. Good, P. Flynn, D. G. Bobrow, and N. Ducheneaut. What a to-do: studies of task management towards the design of a personal task list manager. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 735–742, New York, NY, USA, 2004. ACM.

[13] V. Bellotti, N. Ducheneaut, M. Howard, and I. Smith. Taking email to task: the design and evaluation of a task management centered email tool. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 345–352. ACM, 2003.

[14] V. Bellotti and I. Smith. Informing the design of an information management system with iterative fieldwork. In *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, DIS '00, pages 227–237, New York, NY, USA, 2000. ACM.

[15] O. Bergman, O. Bergman, R. Beyth-marom, and R. Nachmias. The user-subjective approach to personal information management systems. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE AND TECHNOLOGY*, 54:872–878, 2003.

[16] O. Bergman, R. Beyth-Marom, and R. Nachmias. The project fragmentation problem in personal information management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 271–274, New York, NY, USA, 2006. ACM.

[17] R. Boardman, R. Boardman, R. Spence, and M. A. Sasse. Too many hierarchies? the daily struggle for control of the workspace. *MATHEMATICAL MODELS", PROCEEDINGS OF THE ACM CHI'96 CONFERENCE*, pages 406–412, 2003.

[18] R. Boardman and M. A. Sasse. "stuff goes into the computer and doesn't come out": a cross-tool study of personal information management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 583–590, New York, NY, USA, 2004. ACM.

[19] R. G. Capra and M. A. Pérez-Quiñones. Re-finding found things: An exploratory study of how users re-find information. *arXiv preprint cs/0310011*, 2003.

[20] M. Czerwinski, E. Horvitz, and S. Wilhite. A diary study of task switching and interruptions. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 175–182. ACM, 2004.

[21] A. N. Dragunov, T. G. Dietterich, K. Johnsrude, M. McLaughlin, L. Li, and J. L. Herlocker. Tasktracer: a desktop environment to support multi-tasking knowledge workers. In *Proceedings of the 10th international conference on Intelligent user interfaces*, IUI '05, pages 75–82, New York, NY, USA, 2005. ACM.

[22] P. Dubroy and R. Balakrishnan. A study of tabbed browsing among mozilla firefox users. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 673–682, New York, NY, USA, 2010. ACM.

[23] N. Ducheneaut and V. Bellotti. E-mail as habitat: an exploration of embedded personal information management. *interactions*, 8(5):30–38, 2001.

[24] S. Fertig, E. Freeman, and D. Gelernter. Lifestreams: an alternative to the desktop metaphor. In *Conference companion on human factors in computing systems: Common ground*, pages 410–411. ACM, 1996.

[25] V. M. González and G. Mark. "constant, constant, multi-tasking craziness": managing multiple working spheres. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 113–120, New York, NY, USA, 2004. ACM.

[26] B. Hanrahan, G. Bouchard, G. Convertino, T. Weksteen, N. Kong, C. Archambeau, and E. H. Chi. Mail2wiki: low-cost sharing and early curation from email to wikis. In *Proceedings of the 5th International Conference on Communities and Technologies*, C&T '11, pages 98–107, New York, NY, USA, 2011. ACM.

[27] D. A. Henderson, Jr. and S. Card. Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface. *ACM Trans. Graph.*, 5(3):211–243, July 1986.

[28] N. Jhaveri and K.-J. Räihä. The advantages of a cross-session web workspace. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1949–1952, New York, NY, USA, 2005. ACM.

[29] W. Jones and K. M. Anderson. Many views, many modes, many tools ... one structure: Towards a non-disruptive integration of personal information. In *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*, HT '11, pages 113–122, New York, NY, USA, 2011. ACM.

[30] W. Jones, P. Klasnja, A. Civan, and M. L. Adcock. The personal project planner: planning to organize personal information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 681–684, New York, NY, USA, 2008. ACM.

[31] W. Jones, A. J. Phuwanartnurak, R. Gill, and H. Bruce. Don't take my folders away!: organizing personal information to get things done. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1505–1508, New York, NY, USA, 2005. ACM.

[32] V. Kaptelinin. Umea: translating interaction histories into project contexts. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 353–360. ACM, 2003.

[33] D. R. Karger and W. Jones. Data unification in personal information management. *Commun. ACM*, 49(1):77–82, Jan. 2006.

[34] D. Kelly and J. Teevan. *Personal Information Management*, chapter Understanding What Works: Evaluating PIM Tools, pages 190–204. University of Washington Press, 2007.

[35] B. MacIntyre, E. D. Mynatt, S. Voida, K. M. Hansen, J. Tullio, and G. M. Corso. Support for multitasking and background awareness using interactive peripheral displays. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 41–50. ACM, 2001.

[36] T. W. Malone. How do people organize their desks? (extended abstract): Implications for the design of office information systems. *ACM SIGOA Newsletter*, 3(1-2):47–49, June 1982.

[37] G. Mark, V. M. Gonzalez, and J. Harris. No task left behind?: examining the nature of fragmented work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 321–330, New York, NY, USA, 2005. ACM.

[38] D. Morris, M. Ringel Morris, and G. Venolia. Searchbar: a search-centric web history for task resumption and information re-finding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1207–1216, New York, NY, USA, 2008. ACM.

[39] A. Osmani. *Learning JavaScript Design Patterns*. O'Reilly Media, 2012.

[40] M. R. Rajamanickam, R. MacKenzie, B. Lam, and T. Su. A task-focused approach to support sharing and interruption recovery in web browsers. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '10, pages 4345–4350, New York, NY, USA, 2010. ACM.

[41] G. Robertson, M. van Dantzich, D. Robbins, M. Czerwinski, K. Hinckley, K. Risden, D. Thiel, and V. Gorokhovsky. The task gallery: a 3d window manager. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '00, pages 494–501, New York, NY, USA, 2000. ACM.

[42] G. Smith, P. Baudisch, G. Robertson, M. Czerwinski, B. Meyers, D. Robbins, and D. Andrews. Groupbar: The taskbar evolved. In *Proceedings of OZCHI*, volume 3, page 10, 2003.

[43] C. Tashman and W. K. Edwards. Windowscape: Lessons learned from a task-centric window manager. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 19(1):8, 2012.

[44] J. Teevan, R. Capra, and M. Pérez-Quiñones. *Personal Information Management*, chapter How People Find Personal Information, pages 22–34. University of Washington Press, 2007.

[45] S. Voida and E. D. Mynatt. It feels better than filing: everyday work experiences in an activity-based computing system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 259–268, New York, NY, USA, 2009. ACM.

[46] S. Voida, E. D. Mynatt, and W. K. Edwards. Re-framing the desktop interface around the activities of knowledge work. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, pages 211–220, New York, NY, USA, 2008. ACM.

[47] Q. Wang and H. Chang. Multitasking bar: prototype and evaluation of introducing the task concept into a browser. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 103–112, New York, NY, USA, 2010. ACM.

[48] S. Whittaker, T. Matthews, J. Cerruti, H. Badenes, and J. Tang. Am i wasting my time organizing email?: a study of email refinding. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 3449–3458, New York, NY, USA, 2011. ACM.

Appendix A

VT IRB-13-008 Approval Letter

Office of Research Compliance
Institutational Review Board
2000 Kraft Drive, Suite 2000 (0497)
Blacksburg, VA 24060
540/231-4606 Fax 540/231-0959
email irb@vt.edu
website http://www.irb.vt.edu

MEMORANDUM

DATE: March 8, 2013

TO: Manuel A Perez-Quinonez, Ankit Ahuja

FROM: Virginia Tech Institutional Review Board (FWA00000572, expires May 31, 2014)

PROTOCOL TITLE: Contextinator

IRB NUMBER: **13-008**

Effective March 8, 2013, the Virginia Tech Institution Review Board (IRB) Chair, David M Moore, approved the New Application request for the above-mentioned research protocol.

This approval provides permission to begin the human subject activities outlined in the IRB-approved protocol and supporting documents.

Plans to deviate from the approved protocol and/or supporting documents must be submitted to the IRB as an amendment request and approved by the IRB prior to the implementation of any changes, regardless of how minor, except where necessary to eliminate apparent immediate hazards to the subjects. Report within 5 business days to the IRB any injuries or other unanticipated or adverse events involving risks or harms to human research subjects or others.

All investigators (listed above) are required to comply with the researcher requirements outlined at:

http://www.irb.vt.edu/pages/responsibilities.htm

(Please review responsibilities before the commencement of your research.)

PROTOCOL INFORMATION:

Approved As:	**Expedited, under 45 CFR 46.110 category(ies) 6,7**
Protocol Approval Date:	**March 8, 2013**
Protocol Expiration Date:	**March 7, 2014**
Continuing Review Due Date*:	**February 21, 2014**

*Date a Continuing Review application is due to the IRB office if human subject activities covered under this protocol, including data analysis, are to continue beyond the Protocol Expiration Date.

FEDERALLY FUNDED RESEARCH REQUIREMENTS:

Per federal regulations, 45 CFR 46.103(f), the IRB is required to compare all federally funded grant proposals/work statements to the IRB protocol(s) which cover the human research activities included in the proposal / work statement before funds are released. Note that this requirement does not apply to Exempt and Interim IRB protocols, or grants for which VT is not the primary awardee.

The table on the following page indicates whether grant proposals are related to this IRB protocol, and which of the listed proposals, if any, have been compared to this IRB protocol, if required.

Date*	OSP Number	Sponsor	Grant Comparison Conducted?

* Date this proposal number was compared, assessed as not requiring comparison, or comparison information was revised.

If this IRB protocol is to cover any other grant proposals, please contact the IRB office (irbadmin@vt.edu) immediately.

Appendix B

Interview Questions

B.1 Projects

- Tell me how you think about projects and why you set them up?
- Do you work on multiple projects simultaneously?
- Do you find it easy to work on multiple projects?
- How do you decide what to work on next in a project?
- How do you decide which project to work on next?

B.2 Tasks

- Tell me how you use the tasks feature.
- Why do you create tasks for projects?
- Do you use flags for tasks and if so why?
- Do you save URLs with tasks and if so why?
- What do you like most and least about the tasks feature?

B.3 Information Fragmentation

- Do you regularly keep information about your projects in multiple applications (email, dropbox, etc.)?

- If you do, do you use those apps in Contextinator?
- Do you add people to projects to see unread email from them?

B.4 General

- What do you like best and least about Contextinator?
- What other features are missing (or you would like to have) in Contextinator?

Appendix C

User Actions Logged

- Opening project
- Closing project
- Open new project page
- Removing a project
- Creating a new project
- Focusing a project window
- Focusing a non-project window
- Opening project homepage
- Focusing the project homepage
- Adding a person Removing a person
- Opening a tab from project homepage
- Removing a tab from project homepage
- Creating a new bookmark
- Opening a bookmark
- Removing a bookmark
- Opening overview
- Closing overview

- Show flagged tasks in overview
- Show email in overview
- Focusing the overview window
- Opening quick launcher
- Closing quick launcher
- Creating a new task
- Removing a task
- Flagging a task
- Unflagging a task
- Marking a task as complete
- Marking a task as incomplete
- Opening a task URL
- Opening an app from project homepage
- Adding a new app
- Removing an app
- Opening an email from project homepage
- Miscellaneous

Appendix D

Survey

Contextinator Survey

Demographics

Gender

- ◯ Male
- ◯ Female

Profession

- ◯ Graduate Student
- ◯ Undergraduate Student
- ◯ Faculty
- ◯ Staff
- ◯ Other (i.e. Industry)

Age

Projects in Contextinator

Having different ways (e.g. Project Overview, Project Homepage, Tasks) to view projects and work on projects in Contextinator helps me be more efficient.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Having all the project information on the Project Homepage helps me be more focused.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

The Project Homepage is an easy way to access information related to the project.

- ◯ Strongly Agree

- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

With Contextinator, I am able to work on multiple projects simultaneously.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Contextinator is able to handle the number of projects I have.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Apps in Contextinator

Do you have a "label/folder/notebook" for projects in Gmail/Google Docs/Dropbox/Evernote?

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Do you have folders for projects in Dropbox/Google Docs? (Pick one)

- ◯ Always
- ◯ Most of the Time
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

Which apps did you set up for projects in Contextinator? (Check all that apply)

- [ ] Gmail
- [ ] Dropbox
- [ ] Google Docs
- [ ] Evernote
- [ ] Trello

Shortcuts to apps on the Project Homepage make retrieval of project information easier.

- ○ Strongly Agree
- ○ Agree
- ○ Neither Agree nor Disagree
- ○ Disagree
- ○ Strongly Disagree

Getting automatically redirected to the project information in the apps (Gmail, Dropbox, etc.) makes it easy to access the information for a project

- ○ Strongly Agree
- ○ Agree
- ○ Neither Agree nor Disagree
- ○ Disagree
- ○ Strongly Disagree

Defining apps (Gmail, Dropbox, etc.) for projects in Contextinator helps me be more organized in those apps.

- ○ Strongly Agree
- ○ Agree
- ○ Neither Agree nor Disagree
- ○ Disagree
- ○ Strongly Disagree

Email

Do you use labels for projects in Gmail? (Pick one)

- ○ Always
- ○ Most of the Time
- ○ Sometimes
- ○ Rarely
- ○ Never

Which types of labels do you use in Gmail? (Check all that apply.)

Which type of labels do you use in Gmail? (Check all the apply)

- [ ] Project names (e.g. work project#1)
- [ ] Groups of people (e.g. family, work)
- [ ] Role (e.g. Job#1, Class#1)
- [ ] Mailing list (e.g. List#1)
- [ ] Service (e.g. Facebook, Twitter, etc.)
- [ ] Importance (e.g. urgent)
- [ ] I don't use labels
- [ ] Other

I like to see the emails in a project on the Project Homepage instead of opening Gmail.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Viewing project email on the Project Homepage helps me avoid getting distracted by other email in my Inbox.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

I capture emails that I need to act on later as a task.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

I add people to projects in Contextinator to capture email related to a project.

- ◯ Yes
- ◯ No

Tasks in Contextinator

When I open a project, the tasks remind me of what I need to do next on the project.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Having the tasks visible on the Project Homepage helps me focus on the things I need to do on the project.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

I visit the Project Homepage to look at the tasks I need to do in a project.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

I visit the Project Overview to look at the tasks I need to do across projects.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Being able to capture website URLs with a task makes it easier for me to complete the task later.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

I am able to postpone an action (task) until later by capturing the context of the task (e.g. the website where I

I am able to postpone an action (task) until later by capturing the context of the task (e.g. the website where I have to do something)

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Which types of URLs do you usually save with a task? (check all that apply)

- ☐ Email that I need to reply to later
- ☐ Email that contains an attachment related to the task
- ☐ Email that contains information to help me complete a task
- ☐ A website where I have to do something to complete the task
- ☐ A resource needed to complete the task (e.g. a PDF file)
- ☐ Other

Flagging tasks helps me be more effective in my task management.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

I flag tasks to help me focus on the important tasks.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Flagging tasks helps me prioritize things I need to do in a project.

- ◯ Strongly Agree
- ◯ Agree
- ◯ Neither Agree nor Disagree
- ◯ Disagree
- ◯ Strongly Disagree

Flagging tasks helps me prioritize things I need to do across all projects.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

Switching between projects in Contextinator

The ability to quickly switch between projects makes it easier for me to work on multiple projects simultaneously.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

Switching projects makes me lost, so I avoid switching unless it is absolutely necessary.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

Reopening tabs for a project helps me resume working on a project.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

Suspending/Closing a project is easy because I don't have to worry about losing my current work.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

- Strongly Disagree

Resuming a project is easy as I am able to quickly gather where I left off.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

Overall Impression

I would use this tool for my day-to-day work.

- Strongly Agree
- Agree
- Neither Agree nor Disagree
- Disagree
- Strongly Disagree

What other features are missing (or you would like to have)?

What did you like best?

What did you like least?

Appendix E

Evaluation Design

My evaluation design was driven by a set of claims from the existing literature, which were also the basis of the design of *Contextinator*. I started out by grouping together similar claims from the literature into a list of principles. For each of these principles, I determined the data to collect through logging, survey and interview questions. The following is the list of principles and the corresponding data collected. In the survey, all Likert scale questions had the following set of choices: Strongly Agree, Agree, Neither Agree nor Disagree, Disagree and Strongly Disagree, as shown in Appendix D.

E.1 Information is fragmented across different collections (or applications) and clustering together information for a project can be useful.

E.1.1 Survey

- Likert: Having all the project information on the Project Homepage helps me be more focused.
- Likert: The Project Homepage is an easy way to access information related to the project.
- Likert: Viewing project email on the Project Homepage helps me avoid getting distracted by other email in my Inbox.
- Likert: I like to see the emails in a project on the Project Homepage instead of opening Gmail.

E.2 Often there is an overlapping set of folder structures, replicated over each information collection (email, files, calendar, etc.)

E.2.1 Survey

- Do you have "label/folder/notebook" for projects in Gmail/Google Docs/Dropbox/Evernote?
- Do you use folders for projects in Dropbox/Google Docs?
- Which apps did you set up for projects in *Contextinator*?

E.2.2 Logging

- Number of (Gmail, Dropbox, Google Docs, etc.) apps defined for each project.

E.2.3 Interview

- Do you regularly keep information about your projects in multiple applications (email, dropbox, etc.)?

E.3 Common multiple interruptions require an easy way to switch between projects.

E.3.1 Survey

- Likert: The ability to quickly switch between projects makes it easier for me to work on multiple projects simultaneously.
- Likert: Switching projects makes me lost, so I avoid switching unless it is absolutely necessary.
- Likert: With Contextinator, I am able to work on multiple projects simultaneously.

E.3.2 Logging

- Number of times users switched between projects

E.3.3 Interview

- Do you work on multiple projects simultaneously?
- Do you find it easy to work on multiple projects?

E.4 Common multiple interruptions require an easy way to save and restore states of projects.

E.4.1 Survey

- Likert: Reopening tabs for a project helps me resume working on a project.
- Likert: Suspending/Closing a project is easy because I don't have to worry about losing my current work.
- Likert: Resuming a project is easy as I am able to quickly gather where I left off.

E.5 A task list manager should have *in the way* property.

E.5.1 Survey

- Likert: When I open a project, the tasks remind me of what I need to do next on the project.
- Likert: Having the tasks visible on the Project Homepage helps me focus on the things I need to do on the project.
- Likert: I visit the Project Homepage to look at the tasks I need to do in a project.

E.5.2 Interview

- How do you decide what to work on next in a project?
- How do you decide which project to work on next?

E.6 A task list manager should support *informal priority lists*, to ensure completion of near-term priority actions.

E.6.1 Survey

- Likert: Flagging tasks helps me be more effective in my task management.
- Likert: I flag tasks to help me focus on the important tasks.
- Likert: Flagging tasks helps me prioritize things I need to do in a project.
- Likert: Flagging tasks helps me prioritize things I need to do across all projects.

E.6.2 Logging

- Number of times tasks were flagged.
- Number of times flagged tasks were marked as completed.

E.6.3 Interview

- Do you use flags for tasks and if so why?

E.7 Capturing the context while creating a task makes it easier to complete the task later

E.7.1 Survey

- Likert: Being able to capture website URLs with a task makes it easier for me to complete the task later.
- Likert: I am able to postpone an action (task) until later by capturing the context of the task (e.g. the website where I have to do something)
- Likert: I capture emails that I need to act on later as a task.
- Which types of URLs do you usually save with a task?

E.7.2 Logging

- Number of times a task is created with a URL.
- Number of times a task is created with a note.
- Number of times a task with a URL is opened.
- Number of times a task with a URL is marked as completed.

E.7.3 Interview

- Do you save URLs with tasks and if so why?