Memsy: Keeping Track of Personal Digital Resources Across Devices and Services

Matthias Geel(✉) and Moira C. Norrie

Department of Computer Science, ETH Zurich, Zürich, Switzerland
{geel,norrie}@inf.ethz.ch

Abstract. It is becoming increasingly difficult for users to keep track of their personal digital resources given the number of devices and hosting services used to create, process, manage and share them. As a result, personal resources are replicated at different locations and it is often not feasible to keep everything synchronised. In such a distributed setting, the types of questions that users want answers to are: Where is the latest version of this document located? How many versions of this image exist and where are they stored? We introduce the concept of global file histories that can provide users with a unified view of their personal information space across devices and services. As proof-of-concept, we present Memsy, an environment that helps users keep track of their resources. We discuss the technical challenges and present the results of a lab study used to evaluate Memsy's proposed workflow.

1 Introduction

It is now common for users to work with multiple computing devices in their daily lives, often using external storage devices or cloud services as a way of moving digital resources between devices and sharing them with others. In addition, web-based media services and social networking sites are used to not only publish resources, but also manage them and make them accessible on the move. As a result, while the problem of accessing resources from different places has been alleviated, the problem of keeping track of them has been exacerbated.

Since storage space has become extremely cheap, there is a tendency to copy, share and distribute more data than ever, unknowingly making it even harder to keep track of all the copies, versions and variants that we produce [12]. For example, the increased mobility offered by a new range of computing devices has resulted in a greater blurring between the private and working environments of users [5] and they now expect to be able to work on the same document in different locations. To do this, the users need to decide how they will move files around, possibly using a cloud storage service such as Dropbox or an external storage device such as a USB flash drive. In addition to being edited, a document may be copied, renamed or moved between folders and the user often ends with multiple versions in different places. Consequently, users often struggle to remember which is the latest version and where it is stored.

S. Kapidakis et al. (Eds.): TPDL 2015, LNCS 9316, pp. 71–83, 2015.
DOI: 10.1007/978-3-319-24592-8_6

Our goal is to provide users with a global view of their digital resources so they can get answers to questions such as: *"Where is the latest version of a document and on which device/service is it stored? Does a previous version still exist somewhere? Where is the original? How many copies exist and to which, if any, file hosting services have they been uploaded?"*. Importantly, we want to do this without requiring them to change their ways of working. The solution therefore has to be embeddable in standard desktop environments and work alongside existing applications and file management tools.

To achieve this, we propose the concept of a *file history graph* for managing metadata about a user's entire information space, possibly spanning several computing and storage devices as well as online file hosting services. As proof of concept, we have developed Memsy, an environment that helps users keep track of resources and reconcile versions based on the file history graph.

We provide an overview of related research findings and studies in Sect. 2 before proposing a general architecture for keeping track of resources in Sect. 3. We then discuss the technical challenges of implementing such a system in a highly distributed setting along with our Memsy solution in Sect. 4. A lab study carried out to evaluate the Memsy workflow is reported in Sect. 5 before presenting concluding remarks in Sect. 6.

2 Related Work

As the number of personal computing devices has increased, the requirements for information work to be carried out successfully have also changed [5,13,14]. In the study by Dearman et al. [5], they reported that participants found managing information across devices the worst aspect of using multiple devices. They argue that we need to be aware of a user's collection of devices and no longer assume a single personal computer.

Consumer-oriented cloud storage solutions have now become an integral part of many personal workflows as highlighted in recent interviews with 22 professionals from different industries carried out by Santosa and Wigdor [14]. While they enable resources to be readily available on multiple devices, none of their participants regarded the cloud as a complete data solution and data fragmentation remains an important topic even in the context of cloud solutions. Participants explicitly stated that they had difficulties in remembering where in the cloud they put their data. To remedy some of the issues, the authors recommend providing improved awareness and visibility of what is happening in the cloud.

Research into how human memory works has shown that users quickly forget important computing tasks [4]. Elsweiler et al. [7] point out similarities between everyday memory lapses and memory lapses in the context of personal information management and suggest that tools should help users recover from such lapses. Consequently, the ability to track and re-find information in a desktop environment is a highly desirable property for today's information management tools [9]. It has been shown that one effective way of providing memory cues is to use the idea of provenance information [1].

Previous research has mainly explored two strategies for helping users keep track of their resources. The first category of tools aim to provide better search and organisational facilities based on metadata/time [6] or tagging [3]. The second category focus on the associations between resources rather than the properties of the resources themselves and include navigation by associations [2] as well as the exploitation of semantic relations between resources [11]. However, most of these solutions have been designed with the underlying assumption of a single desktop computing environment and it is unclear how they could be adapted to multi-device environments. In fact, the authors of the *Stuff I've Seen* system [6] report from their user study: "The most requested new feature [in the user study] is unified access across multiple machines."

Studies have shown that users are actually very fond of their folder structures [10] as a means of not only categorising information but also decomposing problems/projects into smaller units. The fact that users have different work practices is reflected in how they set up their working environments. For example, some use the desktop as a temporary workspace for all kinds of documents based on how they use their physical desks [15] and cross-device work patterns can often be complex [14].

Since users often put a lot of effort into customising their work environments and file organisation, they are very reluctant to change their work habits. For example, Jaballah et al. [8] proposed a digital library system that enables documents to be browsed via various metadata axes, but, in their diary study, most users preferred the more limited folder view and this was in large part attributed to the time required to become familiar with a new application interface.

Despite their shortcomings, folder hierarchies remain the basis for file organisation. This apparent discrepancy between what has been proposed in research and how people actually work led us to conclude that user behaviour changes very slowly and new solutions should try to build upon existing conceptual models and incrementally refine them. Additionally, we argue that it is crucial to embrace rather than oppress the diversity in strategies and methods people employ to organise information, and take these into account when designing new tools for managing personal resources.

3 Version-Aware Environment

The main idea behind our approach is to build a global provenance index that spans the user's entire personal information space by collecting data from multiple devices and integrating third-party services. Because system access should be independent of any individual computing device, we opted for a classic client-server approach. Figure 1 illustrates the architecture of the Memsy environment. Our system is comprised of three main components which together realise a "version-aware" computing environment for end-users.

Global Resource Catalogue. This is the main service that orchestrates all components and communicates with various client services and tools. It provides a

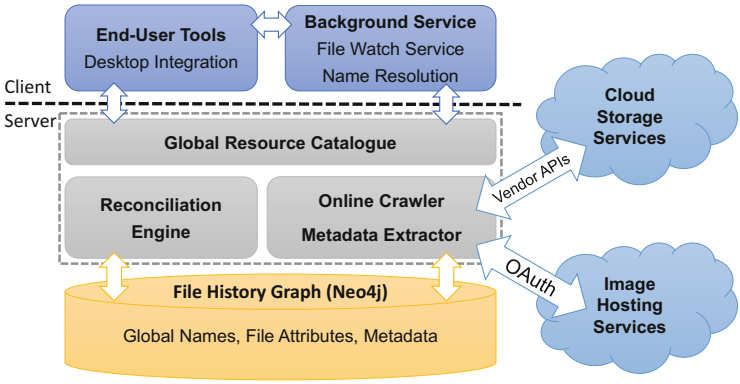

Fig. 1. Architecture for a Version-Aware Environment

global view of a single user's personal information space by ensuring that each resource is not only uniquely identifiable but can also be located in a unambiguous way. A file that is part of the global resource catalogue is called a **managed file**. All other files, including those that are not part of a user's personal information space, are **unmanaged**. The global resource catalogue is responsible for collecting updates from client devices as well as cloud storage and image hosting services (through online crawlers) and applying them to the file history graph.

File History Graph. The file history graph provides a lightweight, implicit versioning mechanism for files. It is essentially an index that stores the metadata required to identify and locate files across devices and services.

Reconciliation Engine. This component reconciles **unmanaged** files, which have not been tracked previously or cannot be tracked reliably (i.e. files on an cloud storage service beyond our control), with existing file histories. In both cases, the goal is to integrate those files into the global resource catalogue. Reconciliation is either done automatically based on file properties or user-driven. In the latter case, we use content-based similarity metrics to aid users in their decisions.

To build a global catalogue of a user's personal information space, each file needs to have a unique and non-ambiguous mapping between a global namespace and its actual location. Figure 2 shows the structure of the global address scheme and an example mapping. Our scheme follows the recommendations for Uniform Resource Identifiers[1]. A vital part of that scheme is the numerical identifier of the storage device on which the file is stored (`storageDeviceId`). In our environment, everything that has a mount point and is writeable by the user can be assigned a globally unique identifier that is then stored in the root directory of that particular storage device. This allows us to reliably recognise external storage devices such as USB flash drives or external harddisks, even when they are moved between computers.

[1] http://tools.ietf.org/html/rfc3986.

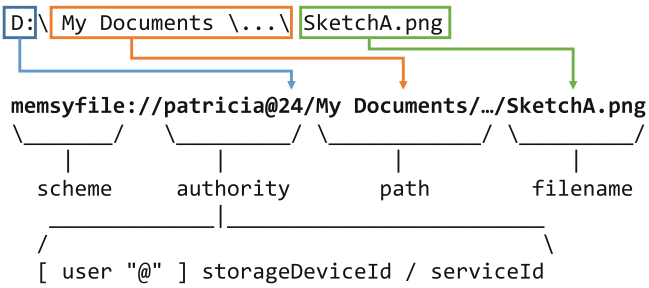

Fig. 2. Global namespace scheme. The example given uses a `storageDeviceId`.

To refer to file hosting services, we use pre-defined names, e.g. *dropbox* or *googledrive*, as identifiers (`serviceId`). The path component corresponds to the original path. In this address translation scheme, we assume that an absolute file path, including the file name, cannot be occupied by more than one file at the same time. This is true for all mainstream file systems (NTFS, ext3, HFS+).

On each computing device belonging to the user's personal information space, a background service is deployed that monitors connected storage devices and tracks the user's file activities. These computing devices also synchronise a list of connected storage devices upon startup and whenever the list changes. This information is used later to tell users about the last known location of "movable" storage devices (such as USB flash drives). Global name resolution at runtime is straightforward because the local service knows the identifiers and mount points of all locally connected storage devices. This is especially useful on the Windows platform, where the mount points (i.e. drive letters) of external storage devices usually depend on the order in which they were connected.

3.1 File History Graph

The file history graph is a data structure that records the metadata of each new version of a file and stores the last known location(s) for each file version. From a data model point of view, each file history is a tree of file version nodes that starts with a single root node. Each version node may point to one or more location nodes. The resulting structure is therefore a forest of directed trees, with each tree representing a separate history.

Figure 3 illustrates how that structure captures file operations. A *create* event triggers the creation of new file history with root V_1 and location L_1. If V_1 already exists, it represents a *copy* operation and we simply add a new location node to the existing node. When a *modify* event occurs, we first identify the (old) file version node that corresponds to the location of the modified file. If that version node is a leaf node, we derive a new version node and re-attach the location node to it. If it is not a leaf, a newer version must already exist somewhere else. In that case, we perform a fork. All other file operations (*rename*, *move* and *delete*) only affect the last known locations of file versions and update them accordingly.

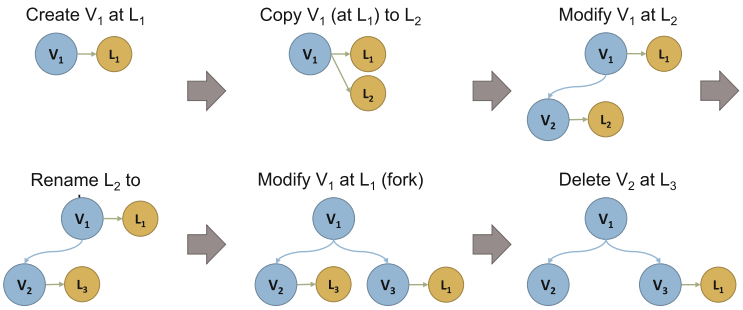

Fig. 3. Evolution of a single file history

As a consequence of the delete operation, file version nodes may end up with zero location nodes attached to them.

It is an important characteristic of our proposed index structure that we retain metadata about previous file versions, even if they no longer exist in the personal information space. With the information in these file histories, we can answer several of our initial questions such as locating copies, latest versions and file origins across devices.

File events are sent immediately to the server if the device is online, otherwise they are cached locally and stored on the storage device where the operations occurred. Thus, the cache travels with the storage device and if, for example, a USB drive is shared between computers that are both offline, the order of executed operations on that USB flash drive is always guaranteed, without the need for complicated time-synchronisation. We have implemented the cache as an event-queue backed by a transactional embedded database (HSQLDB). That queue is consumed by a background thread and entries are only removed if the event could be successfully submitted to server, otherwise processing is halted until Internet connectivity is restored. Since all events (update, move, rename, delete) are idempotent *if* executed in order, we not only achieve at-least-once semantics but can also guarantee that all locally observed transactions are eventually reflected correctly in the global file history graph.

3.2 Reconciliation of Related Resources

At some point, we are bound to encounter resources that either have not been tracked previously or originate from an environment that cannot be monitored directly. We call the process of turning such unmanaged resources into managed (tracked) ones *reconciliation*.

With file hosting services, reconciliation is required as changes to personal resources often cannot be tracked reliably from the outside. Even worse, some image hosting services re-size and re-compress uploaded images, thus changing the actual file content quite significantly. Furthermore, there will always be cases where a file enters a personal environment from *the outside*, for example as an e-mail attachment, downloaded file or copy from somebody else's USB stick.

To align such files with our global resource catalogue, we primarily rely on hashing and file path matching. To recognise copies, we need to be able to tell whether two files have the same content. Popular file hashing functions such as MD5 or SHA1 have proven to be very effective and we therefore use them to realise this functionality. Since we store these hashes for previously encountered versions as well as the current ones, we can reliably reintegrate files that correspond to older versions. For example, a user might have received a report from a co-worker by e-mail, saved it in a local, observed location and modified its content. Several weeks later, they might want to retrieve the revised version but cannot remember where they stored it. Luckily, they remember the e-mail that contained the original file. With the attachment from that e-mail serving as an entry point into the file history, they can learn about the location of the revised version. Having such provenance information allows users to discover newer versions of documents based on the copy of an older one.

To reconcile images that have been modified by online services upon upload (e.g. re-compression on Facebook), computer vision algorithms are used to produce a "perceptual hash". The main idea of such an image signature algorithm is to map an arbitrary image to an arbitrary *sparse image representation* which is significantly reduced in size but retains important perceptual information. An appropriate distance metric is then used to compare these sparse image representations in order to find near-duplicates. So far we have only implemented content-based reconciliation for images, but consider reconciliation to be a vital part of any environment for managing personal resources and plan to extend the service for other types of files.

3.3 End-User Experience

Users first install the Memsy client on their computer(s). Using these clients, they can add local folders to the observed environment. This allows very fine-grained control over which parts of the file system are monitored. Similarly, they can connect and add personal USB flash drives. Such storage devices can also be labelled to make it easier to recognise them when they show up in search results. After setting up the environment, users can continue to work as usual, freely creating, moving, renaming, copying and deleting files as well as creating new folder structures within the monitored folders.

If a user wants to find the latest version of a file, they follow a simple procedure shown in Fig. 4. (1) Small overlay icons in the file explorer view offer a quick glance of the current file status. From a user experience point of view, this works similarly to overlay icons in desktop integrations of popular cloud storage services (e.g. Dropbox) or version control systems (e.g. TortoiseSVN). A tick symbol indicates that the current file is up-to-date, while an exclamation mark warns the user that a more recent version is stored somewhere else. Such icon overlays give a simple and unobtrusive way of providing status information to users. They integrate well with the desktop experience, work in all standard file dialogues and are well-suited to handling directories with a large number of

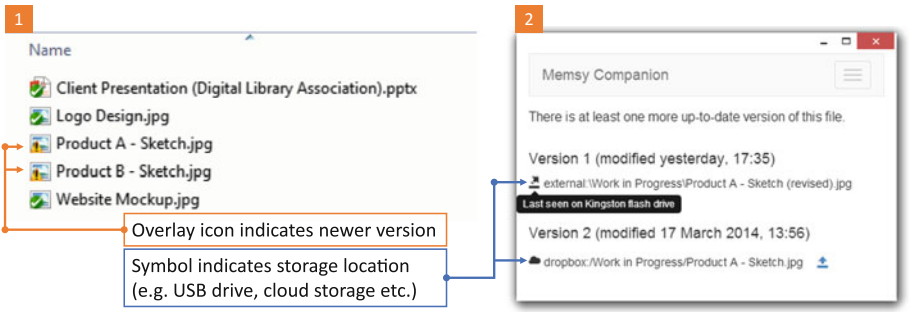

Fig. 4. A Version-Aware Environment: (1) Overlay icons inform user about file status [cropped screenshot from Windows file explorer]. Context menu entry invokes custom application (2) with further information about related versions.

files. When the user right-clicks one of those files, a few Memsy-specific commands appear in the contextual menu. (2) From there, the user can launch the Memsy Companion application, a small desktop tool that displays information retrieved from the global resource catalogue. In our example, the user learns that there are two newer versions located on other devices/services. We can show for each version whether that particular file can be accessed directly. Hovering over entries that are currently not available provides information about its last known location (a USB flash drive in this example).

4 Implementation

To demonstrate and evaluate our approach, we have developed a reference implementation of the Memsy environment in Scala, a JVM-based, multi-paradigm programming language that offers functional and object-oriented constructs. The global resource catalogue called *Memsy Global* runs on a dedicated server. It is built with a micro web framework called Scalatra[2]. As a backend for the file history graph, we use Neo4j[3], a graph database implemented in Java, which is a perfect fit for tree structure of our file history graph. On the client-side, we deploy a background service called *Memsy Local*, which is responsible for monitoring the users' file activities. It communicates with the server using its REST-style API over HTTP, with JSON as the data exchange format. To capture file system events, we make use of the Java 7 File API (NIO.2). To incorporate a number of popular online services, we have implemented crawlers for Dropbox, Google Drive, Facebook and Flickr. The former two cloud storage crawlers process all supported files whereas the latter two only retrieve and process images.

The user interface has been implemented for the Windows platform using a number of high- and low-level technologies. Both icon overlays and the context

[2] http://www.scalatra.org/.
[3] http://www.neo4j.org/.

menu have been realised as Windows shell extensions. To indicate a file's status, an icon overlay handler was created for the Windows file explorer that queries the local Memsy client, which in turn may have to ask Memsy Global for the current status. Because icon overlays are precious resources in Windows (only up to 16 handlers can be registered), our integration builds upon TortoiseOverlays, the icon handler of many popular desktop integrations of version control systems such as Subversion, Git or Mercurial. This makes the icons easily recognisable by users of any of these solutions.

Our dedicated end-user tool, the Memsy Companion app, is a Chrome app. These apps resemble native desktop applications but run on top of Google Chrome and can be built using web technologies. The tool communicates with the local background service and the global resource catalogue to answer the questions motivated earlier. We have chosen this web technology-centric approach because it allows us to seamlessly mix various technologies from high-level web interfaces down to low-level shell extensions. The consistent use of the HTTP protocol also facilitates similar integrations for other operating systems.

5 Evaluation

To evaluate our approach, we conducted a small user study with two primary goals. First, we wanted to gather valuable insight into work practices and strategies for keeping track of resources across different devices. Second, we wanted to find out whether the tool support would be sufficient when a user cannot remember what happened to a particular document.

The main study design challenges were: *How can we reliably create a condition that reflects the situation where a user has actually forgotten the location of the latest version? How can we eliminate non-controllable factors such as an individual's particular cognitive abilities?* For these reasons, we decided to perform a controlled lab experiment instead of an external field study.

Study Setup. We set up three work stations in a room to represent different work places. Each work station had a labelled USB stick. All 6 devices were configured to belong to the same user and all tools were pre-installed on the workstations. We prepared a set of 20 files for our hypothetical Memsy user: 5 presentations (PowerPoint), 5 reports (Word) and 10 images (JPEG). These files had been copied to a managed folder on each workstation.

Participants. Our participants were recruited from an interdisciplinary European research project where our group is a member. This allowed us to recruit not only computer science researchers (3 professors, 3 research assistants), but also people from media education/pedagogy (4), plus a designer and an architect. In total, we had 12 participants (half of them female), from industry as well as from academic institutions. Although we had a rather small group of participants, they were from 5 different organisations with quite different work environments and this matched our belief that a diverse set of study subjects would yield more interesting results than a large, but homogeneous, subject pool.

Procedure and Tasks. We divided our 12 participants into groups of 3 persons. All participants of a group were invited at the same time and each participant was randomly assigned to a designated work station. The study started with a pre-task questionnaire that collected background information about participants' experiences with file managers and their current techniques for version management of documents and copying files between devices.

The main study was a sequence of information management tasks, where each participant performed exactly the same steps but worked on a different subset of the files. In the first part of the study, participants were asked to perform three common file management and document editing tasks: update and rename a presentation, copy an image to a (newly created) subfolder and modify it, copy a report to the USB drive and modify it. Though we asked participants to follow our instructions closely, we did not tell them how to accomplish the tasks. It was up the individual user to decide whether they work with keyboard shortcuts, drag & drop or context menus. If they made a mistake (e.g. copied the wrong file), we asked them to revert the change how they saw fit. At that stage, the background service monitored the file management operations behind the scenes and propagated the changes to the global resource catalogue.

In the second part, participants were asked to gather information for an upcoming presentation which included files possibly modified by the other participants (1 presentation, 1 report, 2 images) and spread across the other two work stations and USB sticks. To complete the task, they had to indicate if there was a newer version of any of those files, and, if there was, where the newer versions were stored (location, drive and path). At any time, they could invoke the Memsy Companion app, shown in Fig. 4, for further information.

Results. In our pre-task questionnaire, we asked participants about their current work practices for managing versions and copying files between devices. In both questions, multiple answers were possible. By far the most popular method for managing multiple versions of office documents is a personal file naming scheme (10 participants). Although less popular, folder structures were still used by half of the participants. Although revision control systems are widely used to manage source code, our results indicate that they are considerably less popular for managing versions of documents (4), let alone to share files between devices (2). None of our participants used a dedicated document management system such as SharePoint.

Interestingly, even though all 12 participants use cloud storage services to move files between devices, almost none (2) use the implicit versioning provided by some cloud providers. This might be due to the fact that popular services such as Dropbox and Google Drive by default only store previous versions for up to 30 days. We conclude that cloud storage services have become a ubiquitous tool for transferring files between machines, but other means such as USB sticks (11 participants) and sending e-mail to yourself (11) still have their uses. These results affirm some of the assumptions underlying our initial use case and further inform our proposed approach.

All participants were able to use the Memsy Companion app to answer most of the questions from the second part correctly (overall completion rate was 46 out of 48 tasks). Figure 5 shows the results from the post-task questionnaire. For each of the statements, we asked participants to rate their level of agreement on a 7-point Likert scale, where 1 is *strongly disagree* and 7 *strongly agree*.

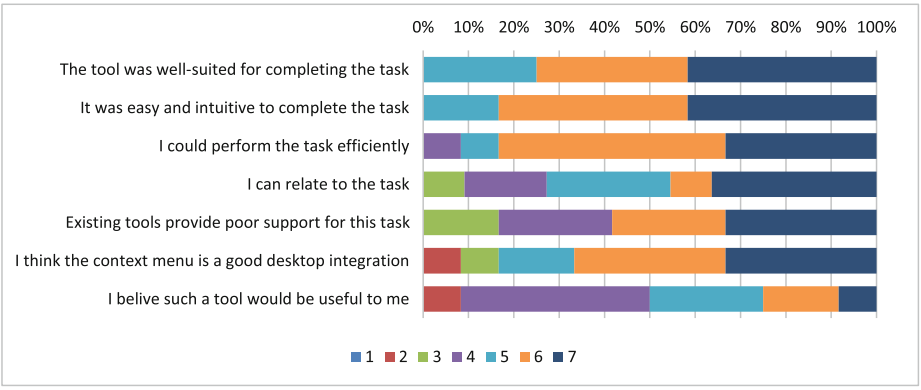

Fig. 5. Results from post-task questionnaire

The Memsy Companion app as well as the entire experience offered by our environment received fairly positive ratings. Most participants felt it was well-suited to the task at hand (median 6), that it was easy and intuitive to complete the task (median 6), and that they were efficient in doing so (median 6). Participants also agreed with our underlying assumption that existing tools [offered by the operating system] provide poor support for the task (median 6). However, while many participants could relate to the task (median 5), most users (mode 4) were undecided about whether such a tool would actually be useful for them. These general concerns were also reflected in some of the feedback we received, for example P3 noted: "*The general issue is what do I actually do when I realise that I don't have the most recent version in front of me.*"

We conclude that, while our prototype is successful in answering the kind of questions posed at the beginning, users may require more information to help them decide what to do when confronted with multiple versions on different devices. As P9 pointed out, "*[The tool should provide] some overview of what kind of change has been done*" and P10 agreed by saying "*[The tool should] provide the differences of versions (visual comparison of versions)*".

With regard to desktop integration, reactions from users were quite positive (median 6) but some expressed a preference for an even deeper integration with the regular desktop computing environment. To that end, we plan to experiment with other forms of shell extensions and consider providing more detailed information as part of the regular file explorer interface. However, further studies are required to steer development in the right direction.

6 Conclusion

We have shown how a global resource catalogue together with a file history graph can be used to help users keep track of personal resources across devices and services. Our aim was to provide a solution that could integrate with existing work practices and applications rather than replace them and our Memsy prototype has demonstrated how this can be achieved, although our initial studies have shown that providing more information about the nature of changes made to files and deeper integration would be desirable and this is something to explore in future research. Further, we have only considered single user environments, but it would be interesting to investigate the implications of multi-user environments where personal resources can be shared with other Memsy users.

References

1. Blanc-Brude, T., Scapin, D.L.: What do people recall about their documents?: implications for desktop search tools. In: Proceedings IUI 2007, pp. 102–111. ACM (2007)
2. Chau, D.H., Myers, B., Faulring, A.: What to do when search fails: finding information by association. In: Proceedings CHI 2008, pp. 999–1008. ACM (2008)
3. Cutrell, E., Robbins, D., Dumais, S., Sarin, R.: Fast, flexible filtering with phlat. In: Proceedings CHI 2006, pp. 261–270. ACM (2006)
4. Czerwinski, M., Horvitz, E.: An investigation of memory for daily computing events. In: Faulkner, B., Finlay, J., Détienne, F. (eds.) People and Computers XVI-Memorable Yet Invisible, pp. 229–245. Springer, London (2002)
5. Dearman, D., Pierce, J.S.: It's on my other computer!: computing with multiple devices. In: Proceedings CHI 2008, pp. 767–776. ACM (2008)
6. Dumais, S., Cutrell, E., Cadiz, J., Jancke, G., Sarin, R., Robbins, D.C.: Stuff I've seen: a system for personal information retrieval and re-use. In: Proceedings SIGIR 2003, pp. 72–79. ACM (2003)
7. Elsweiler, D., Ruthven, I., Jones, C.: Towards memory supporting personal information management tools. J. Am. Soc. Inf. Sci. Technol. **58**(7), 924–946 (2007)
8. Jaballah, I., Cunningham, S.J., Witten, I.H.: Managing personal documents with a digital library. In: Rauber, A., Christodoulakis, S., Tjoa, A.M. (eds.) ECDL 2005. LNCS, vol. 3652, pp. 195–206. Springer, Heidelberg (2005)
9. Jensen, C., Lonsdale, H., Wynn, E., Cao, J., Slater, M., Dietterich, T.G.: The life and times of files and information: a study of desktop provenance. In: Proceedings CHI 2010, pp. 767–776. ACM (2010)
10. Jones, W., Phuwanartnurak, A.J., Gill, R., Bruce, H.: Don't take my folders away!: organizing personal information to get things done. In: Proceedings CHI 2005 Extended Abstracts, pp. 1505–1508. ACM (2005)
11. Karger, D.R., Bakshi, K., Huynh, D., Quan, D., Sinha, V.: Haystack: a customizable general-purpose information management tool for end users of semistructured data. In: Proceedings CIDR (2005)
12. Karlson, A.K., Smith, G., Lee, B.: Which version is this?: improving the desktop experience within a copy-aware computing ecosystem. In: Proceedings CHI 2011. ACM (2011)

13. Oulasvirta, A., Sumari, L.: Mobile kits and laptop trays: managing multiple devices in mobile information work. In: Proceedings CHI 2007, pp. 1127–1136 (2007)
14. Santosa, S., Wigdor, D.: A field study of multi-device workflows in distributed workspaces. In: Proceedings UbiComp, pp. 63–72. ACM (2013)
15. Zacchi, A., Shipman, F.M.: Personal environment management. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 345–356. Springer, Heidelberg (2007)