Context Spaces as the Cornerstone of a Near-Transparent & Self-Reorganizing Semantic Desktop

Christian Jilek[1,2], Markus Schröder[1,2], Sven Schwarz[1], Heiko Maus[1], and Andreas Dengel[1,2]

[1] Smart Data & Knowledge Services Dept., DFKI GmbH, Kaiserslautern, Germany
[2] Computer Science Dept., TU Kaiserslautern, Germany
{firstname.lastname}@dfki.de

Abstract. Existing Semantic Desktops are still reproached for being too complicated to use or not scaling well. Besides, a real "killer app" is still missing. In this paper, we present a new prototype inspired by NEPOMUK and its successors having a semantic graph and ontologies as its basis. In addition, we introduce the idea of context spaces that users can directly interact with and work on. To make them available in all applications without further ado, the system is transparently integrated using mostly standard protocols complemented by a sidebar for advanced features. By exploiting collected context information and applying Managed Forgetting features (like hiding, condensation or deletion), the system is able to dynamically reorganize itself, which also includes a kind of tidy-up-itself functionality. We therefore expect it to be more scalable while providing new levels of user support. An early prototype has been implemented and is presented in this demo.

Keywords: semantic desktop · context · transparent integration · self-reorganization · managed forgetting

1 Introduction

After its hype finally receded about half a decade ago, rather few advances in Semantic Desktop (SemDesk) research have been reported. An overview of (modern) SemDesks can be found in [1]: Existing implementations are, for example, reproached for being rather complicated to use, not scaling well (thus draining lots of system resources), and there is still no real "killer app" available. Concerning SemDesk applications, two categories could be observed: newly created semantic applications and plug-ins to enhance traditional, non-semantic ones [1].

As a successor to the *NEPOMUK Semantic Desktop*[3], DFKI's Smart Data & Knowledge Services department developed its own prototype[4] [5] making

[3] `www.semanticdesktop.org`

[4] This prototype is permanently used in the department since 2012, currently having a group knowledge graph of approx. 2.6 million triple statements.

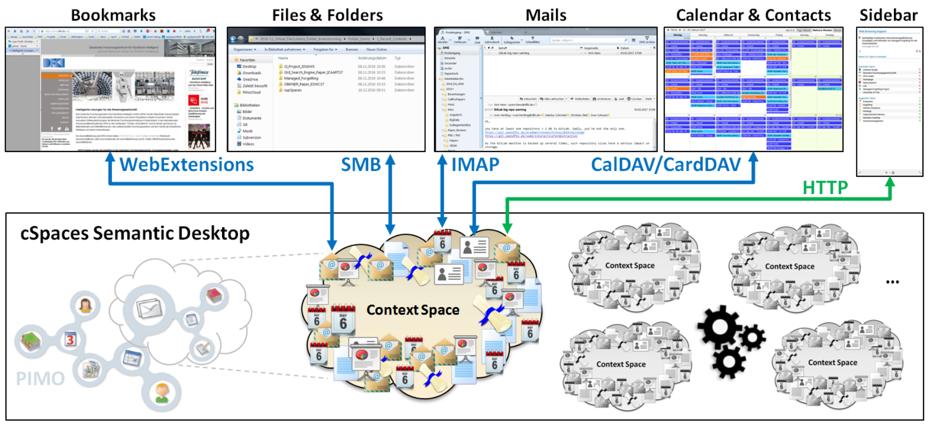

Fig. 1. Conceptual overview of the cSpaces Semantic Desktop

SemDesk technology ready for 24/7 usage in practice, covering private and corporate scenarios. After lessons learned in various projects, we now propose *Context Spaces* as an extension of this prototype addressing the issues mentioned before.

2 Approach

Context Spaces. One of SemDesk's cornerstones is the Personal Information Model (PIMO) [7], which tries to represent a user's mental model as well as possible (especially in a machine understandable way). Information items (files, mails, bookmarks, ...) that are related to each other in a person's mind, but are separated on their computer (file system, mail client, web browser, ...), can thus be interlinked. With *Context Spaces* (or *cSpaces* for short) we extend this idea by explicitly (and additionally) associating items with contexts of the user (see lower left of Figure 1). This is based on the intuition that every activity is performed in a certain context. Hence, each information item stored on a person's computing device can be associated with one or more contexts (association strength may vary depending on the user's current context awareness). We therefore assume that users are explicitly aware of the concept of context [2] and that they are also aware of their current context (at least most of the time). Examples of such contexts are: *Spain holiday 2017*, *prepare ESWC18 paper*, or *my childhood memories*. We do not enforce a certain definition of context: users should be able to stick to their own conceptualization as much as possible. However, we do assume that contexts express a certain relatedness of its elements. Besides being a kind of container for things, they may also be strongly related to (calendar) events or tasks. Our context model extends [9], e.g. by introducing context hierarchies or forgetting aspects. Instead of just having context as passive metadata, we in addition see it as an accessible element users can interact with (create new (sub)contexts, split or merge them, add/remove elements, etc.).

SemDesk user studies [8] revealed that people omitted rather specific relations in favor of basic ones (like *isRelatedTo* or *isPartOf*), whereas the system is formal where possible, e.g. representing calendar events or address book contacts. This matches our idea of providing a low effort opportunity to keep things a bit more tidied up when simply associating them with a certain context (or multiple contexts). Additionally, some of these associations may also be inferred by the system reducing manual effort even more, e.g. a received email reply can automatically be associated with the original mail's context. More advanced features supporting the user will be discussed in the section after next.

Transparent Integration. Using contexts as an explicit interaction element only makes sense if applications also respect them. Like illustrated in Figure 1, we therefore integrate cSpaces into the rest of the system using standard protocols like *Server Message Block (SMB)* for files, *IMAP* for mails, *CalDAV* for calendar entries and tasks, and *CardDAV* for contacts. For web browsers, we use *WebExtensions*[5], which provide cross-browser functionality and an integration level similar to having an underlying protocol. Thanks to these well-supported standards, we are able to inject our own file/mail/bookmark/calendar system deep into the surrounding system without modifying its code. Applications then operate on information items, which are actually parts of the knowledge graph (PIMO) transparently provided to them by the cSpaces app. Especially in corporate scenarios, it is very convenient if users may just work with the resources in their contexts without caring whether they are actually spread across various sources like intranet shares, for example.

Utilizing only standard protocols has certain limitations due to their rather basic, low-level character. Some activities, like writing a note or comment about a resource, can become inconvenient or non-intuitive. To avoid this, we provide an additional sidebar as a single interaction point for using advanced features. Users therefore do not need to learn a new (plugged-in) interface for each of their applications. They can just keep using them the usual way having only the sidebar as a new UI to become familiar with. From the development point of view, the effort of creating and maintaining plug-ins needed for higher level functionality is comparatively low to that of earlier SemDesks. They can be realized as *headless plug-ins* having very little functionality, often just the capability of *sending out* in-app events to the sidebar (that is why we also shortly call them "plug-outs"). In addition, their corresponding UI elements and logic are located in the sidebar, where they can be easily reused. Plug-outs for different mail clients could, for example, share the same tagging UI.

Self-Reorganization. Features discussed so far primarily aim at our system's ease of use. The other aspects mentioned in the beginning (scalability, missing "killer app") will be addressed using *Managed Forgetting*, by which we understand an escalating set of measures: temporal hiding, condensation, adaptive synchronization, archiving and deletion of resources and parts of the knowledge graph [6]. By having users work on cSpaces, we gather rich contextual information about all of their resources, which allows the system to semi-automatically help them in organizing their stuff. Thus, cSpaces are continuously spawned, retracted, merged, condensed, or forgotten. Some evidence for this is collected

[5] `https://wiki.mozilla.org/WebExtensions`

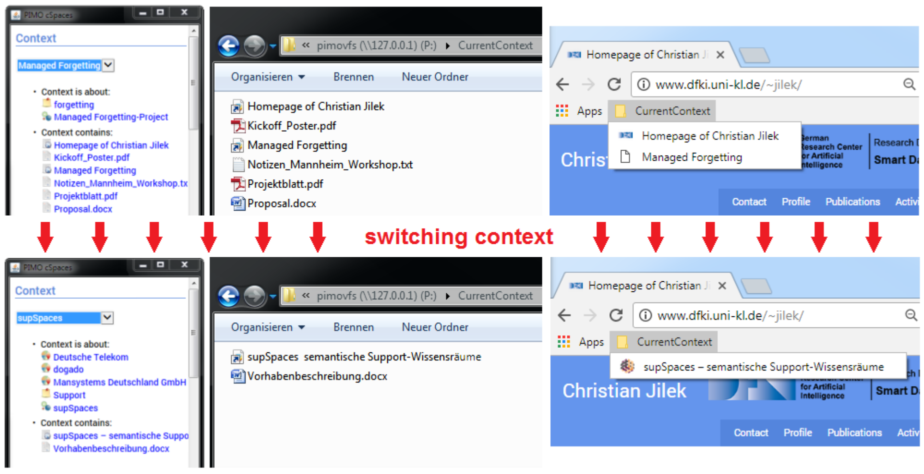

Fig. 2. Screenshot of sidebar, file explorer and browser before (top) and after a context switch (bottom), illustrating the effects of a dynamic reorganization of the system.

automatically (plug-outs, protocol logs), other hints are explicitly given by the user (create/modify cSpaces, tagging, etc.). As an example, let us assume we do a consulting job for company XY. The contract involves five meetings about different topics. Our system could represent this by having an overall cSpace containing general information about XY, e.g. contact and contract information. For each meeting, there could be an individual sub-cSpace about its respective topic. Several months after the job has been completed, the system starts to remove details, e.g. train schedule to get to the meeting or auxiliary material for doing the presentation. After some years have passed, the sub-cSpaces could be merged with their parent, since the separation into different meetings is not relevant anymore. Only the most important items, e.g. individual reports or an overall final report, are kept. All other items are either condensed, moved to an archive or deleted completely (which can be adjusted by the user on a general level). An item's current and estimated future value for the user are therefore continuously assessed resulting in different forgetting measures like temporal hiding (e.g. some items during one of the meetings), deletion, etc. This especially means that the system is able to reorganize itself to a certain extent, which especially includes a kind of tidying-up-itself functionality. Some of the described features have already been implemented and successfully used in our research and industry prototypes [3,4], however most of them are still under heavy development.

Demo. In an early proof-of-concept implementation based on [5], we already realized some of the file system, browser and calendar parts. The screenshots in Figure 2 show a typical feature of our system: the user selects a different context using the sidebar. As a consequence, the *current context*, available as a folder in the file system as well as the browser, is dynamically reorganized by our app. Note that the system tries to present meaningful views on the current context in each app: e.g., the view in the browser only contains web links. To really get an impression of how the interaction with the system looks like, we kindly refer the reader to our demo video[6], which also shows some additional features.

3 Conclusion & Outlook

In this paper, we presented a new SemDesk prototype based context spaces that users directly interact with and work on. The system is transparently integrated using mostly standard protocols complemented by a sidebar for advanced features. Users may thus stick to their favorite applications which should strongly contribute to the overall ease of use. Learning efforts are presumably low due to the sidebar being the only new UI that is introduced. By exploiting its collected context information and applying features of Managed Forgetting, the system is able to dynamically reorganize itself which also includes a kind of tidying-up-itself functionality. We therefore expect it to be more scalable than its predecessors while providing new levels of user support. Nevertheless, a lot of functionality still needs to be fully implemented and evaluated. We plan to do extensive user studies once the system matures.

Acknowledgements. This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – DE 420/19-1.

References

1. Dragan, L., Decker, S.: Knowledge management on the desktop. In: EKAW, 2012. pp. 373–382 (2012)
2. Gomez-Perez, J.M., Grobelnik, M., Ruiz, C., Tilly, M., Warren, P.: Using task context to achieve effective information delivery. In: 1st Workshop on Context, Information and Ontologies. pp. 3:1–3:6. CIAO '09, ACM (2009)
3. Jilek, C., Maus, H., Schwarz, S., Dengel, A.: Diary generation from personal information models to support contextual remembering and reminiscence. In: 2015 IEEE Int'l Conf. on Multimedia & Expo Workshops, ICMEW 2015. pp. 1–6 (2015)
4. Jilek, C., Schwarz, S., Maus, H., Dengel, A.: Managed forgetting, data condensation & preservation in application. In: 2016 ACM Int'l Joint Conf. on Pervasive and Ubiquitous Computing: Adjunct. pp. 1046–1053. UbiComp '16, ACM (2016)
5. Maus, H., Schwarz, S., Dengel, A.: Weaving personal knowledge spaces into office applications. In: Fathi, M. (ed.) Integration of Practice-Oriented Knowledge Technology: Trends and Prospectives, pp. 71–82. Springer (2013)
6. Maus, H., Jilek, C., Schwarz, S.: Remembering and forgetting for personal preservation. In: Mezaris, V., Niederée, C., Logie, R.H. (eds.) Personal Multimedia Preservation: Remembering or Forgetting Images and Video, pp. 233–277. Springer (2018)
7. Sauermann, L., van Elst, L., Dengel, A.: PIMO – a framework for representing personal information models. I-Semantics 7, 270–277 (2007)
8. Sauermann, L., Heim, D.: Evaluating long-term use of the gnowsis semantic desktop for PIM. In: The Semantic Web - ISWC 2008. pp. 467–482 (2008)
9. Schwarz, S.: A context model for personal knowledge management applications. In: Modeling and Retrieval of Context, 2nd Int'l Workshop, MRC 2005, Revised Selected Papers. pp. 18–33 (2005)

[6] find this paper's demo website at `https://pimo.opendfki.de/cSpaces/`