Project InfoSpace: From Information Managing to Information Representation

Pamela Ravasio, Ljiljana Vukelja, Gabrio Rivera & Moira C. Norrie

Swiss Federal Institute of Technology, 8092 Zurich, Switzerland

{vukelja|rivera|norrie}@inf.ethz.ch ; ravasio@iha.bepr.ethz.ch

Abstract: The Desktop metaphor has repeatedly been declared antiquated in various ways. It nevertheless continues existing. Despite the widely pronounced critique, it is still not clear what functional requirements have to be met by a new none-desktop interface. In this context, we define and explain the functional requirements (paradigms) we think fundamental for upcoming generations of non-desktop user interfaces. In close collaboration with users, we have also developed a graphical user interface (GUI) that illustrates these new paradigms and which is at the same time a concrete proposition for a new metaphor.

Keywords: Desktop metaphor, user requirements, information paradigms, prototype, abstract metaphor.

1 Introduction

The Desktop metaphor continues to exist despite the fact that it is widely agreed to be antiquated. In the past decade, a number of approaches have been undertaken in order to address its known drawbacks. Lifestreams (Freeman et al., 1995; Fertig et al., 1996) for instance, tried to overcome the problems of: rigidity of the directory hierarchy; location dependence of information access; non-transparency of information storage due to compulsory file and folder naming; manual archiving by the user. The Graphical User Interface (GUI) designed for the system gave access to a time-ordered (diary-like) document view to the document currently in use and to an overview of metadata information about the document itself. TimeScape (Rekimoto, 1999) continued further on this trail. It presented a GUI based on the idea of document evolution tracking, i.e. the 'desktop' could be scrolled backwards and forwards in time like a time-machine, and documents appeared and disappeared accordingly in their corresponding versions. Finally, Placeless Documents (Dourish et al., 1999; Dourish et al., 2000) abandoned any form of a structured document space, and replaced it with a concept of placeless documents with attached properties. The corresponding GUI offered a properties based drag-and-drop document query mechanism, but was otherwise based on the traditional desktop metaphor.

Recapitulating and completing the requirements formulated in earlier efforts, any new personal information management system must meet the following user requirements (functionalities):

1. Unified handling of all types of documents (text, pictures, emails, etc.)
2. Multiple classification of documents, i.e. in various contexts, and therefore context dependent behaviour.
3. Bi-directional links between documents in order to manage semantic associations and access related information directly.
4. Tracing of documents' temporal evolution.
5. Transparent physical location of a particular piece of information.

This list summarises and extends requirements partially formulated and addressed in earlier efforts (1 ,4, 5) as well as additional ones (2, 3).

In this paper, we present a GUI that was developed in a top-down approach in order to give the users access to these functionalities.

Section 2 introduces the system sided principles and paradigms needed to meet the above requirements. Section 3 describes the process followed during the design and evaluation phase. Section 4 then explains how the designed GUI implements the paradigms explained in Section 2. Finally, Section 5 wraps up the experiences we made, the users' reactions to their first time hands-

on experiences with the new functionality and the design; future work is also pointed out.

2 Information Model and Paradigms

Information management, organisation and representation are central activities in today's use of computers and much of that information is embodied in documents. We consider email as a significative example for exposing our motivations. Email has become a major channel for the dissemination of information and email tools have nowadays attained a dominant information management role. Yet, from both information management and information representation perspectives, most email tools are still very primitive. Some limitations have their roots in the core information meta-model itself. They manifest themselves through the inability to support rich and flexible means of organising and interacting with emails: Rigid and hierarchical classification structures are a direct consequence.

Emails are indeed a good example for showing a typical way of how users handle information. Emails can have different types (MIME), can be classified in different manners, depending on the sender, receiver, subject, date or content and they may be associated with other mails. For instance, a reply can be associated with the received mail. The classification and association mechanisms have to be dynamic, so that, at any time, new structures can be introduced within the system. Note that the use of these three features, namely typing, classification (or categorisation) and dependencies (or associations) can be generalised to all kinds of documents and are not restricted to emails only. Hence, all other applications may potentially take advantage of this semantic information.

The OM model (Norrie et al., 1996) introduces the three above mentioned constructs and is a well-suited model for information management and structuring. In fact, it supports both a rich, flexible and type independent classification mechanism, as well as dependencies between objects. The Universal Information Platform (UIP) is an object-oriented, multi-user, distributed, persistent information management system (Rivera et al., 2002) based on the OM model. The UIP provides a complete and rich application programming interface (API) which can be used for a range of application environments built on the top of the UIP itself.

As soon as we tackled the task of developing a suitable GUI for the UIP, in order to manage, organise and retrieve (any kind of) information, we were confronted with the problem of how to represent these three basic features of the OM Model appropriately.

The first key issue is the representation of different contexts (*types*) for the same document. A document accessed under the context of an *html type* can have different attributes and functionality as the same document accessed under the context of a *plain-text type*. In the first case (*html type*) for instance, the execution of a *print method* might produce a paper-copy of the rendered document, while in the second case (*plain-text type*), the method execution would print out the raw html source.

The second key issue is the representation of different categorisations. Documents can be classified in different ways, independently of their respective types. It should be possible to have a hierarchical classification as well as a flat one if the user wishes so. It should also be possible to create new dynamic classifications, for instance as a result of a query to the system. All these classifications have the right to co-exist concurrently. In other words, the challenges here for a GUI are: (1) in a user-convenient way, make them aware of all the classification structures available for a particular document; (2) assist the user in the task of organising her information.

Finally, the third and last key issue is the representation of the dependencies between documents. A document may be associated with several others, according to specific semantics (e.g. *document contains picture)*. The question here is how to represent this association as well as how to support the navigation from one document to the other(s), based on the linking mechanism.

3 Design Process

Our work began with identifying the key operations that users could/should perform with the system. Initially, we defined 17 central tasks which would have to be accomplishable using the prototype. These tasks encompassed finding, opening, or saving operations, as well as more demanding actions such as creating links between individual documents or document groups, or searching the system for a determined document manually or with the help of a search tool.

Three metaphors were prototyped in a paper and cardboard fashion. Each metaphor represented a different abstraction level: (A) Concrete: A physical 3D filling system with drawers. (B) Medium: Lenses with overview and zoom functionality. (C) Abstract: A graph based representation.

An interaction designer, an HCI specialist and a computer scientist then verified each design using Heuristic Evaluations. Iteratively, the three designs were improved and re-verified, until they were considered consistent and faultless.

At this stage, we decided to go on with prototype C. The reasons for this decision were: (1) We wanted to ascertain if this approach - using an abstract instead of a concrete metaphor - would be accepted by our users. (2) We expected fundamental user input based on this first prototype, which then should be directly fed back into any further development of prototypes A and B. Paper and cardboard prototype C was hence converted into a PowerPoint prototype, which again was verified by the three experts for its consistency and correctness.

In a usability study, the PowerPoint prototype was then exposed to users who were asked to execute the 17 previously defined tasks. The users (3 medium skilled users; 2 expert users familiar with the UIP) were asked to judge the prototype with respect to: functionality, subjective satisfaction for task completion and ease of task execution. Each user session was recorded on video tape. Users' comments, problems and suggestions were then extracted from the tapes, summarised for each individual task and finally compared task-wise against each other. A list of improvements, changes and conclusions was compiled. At the time of writing, the list is used for the implementation of a fully functional version of prototype C, and for the improvement of the remaining two designs in their early development stages.

4   Research prototype

In this section we explain the look-and-feel (Figure 1) of the designed prototype and the way it implements the paradigms outlined in Section 2.

Figure 1: The look-and-feel of our prototype.

4.1   Categories and Contexts

A category represents a collection of documents (e.g. individual Emails) that are of a specific type (e.g. Email). As a consequence for the user, who is dealing with entire collections *and* therein contained individual documents, three aspects are of importance: overview of the entire available information space in terms of available categories; access to the contents of an individual document or collection needed for working; awareness that a particular document may belong to several of the existing categories at the same time.

The first issue is dealt with by the *Overview area* (A). It serves for the user's navigation in her personal information space. In the style of the system's data model (Norrie et al., 1996), a directed, acyclic graph (DAG) of nodes and two types of edges is used to represent the whole of a user's personal information space. Therein, *circles* represent collections of documents. Their role corresponds to the one of folders in today's desktop metaphor. *Straight lines* between circles are used for super/sub collection relationships. *A double circle* indicates the collection in which the on-going user action (e.g. editing of a file) is currently underway. W*hite colour* is used for the sub graph used for work in progress, whereas *light grey* is used for the remaining, non-activated parts of the graph. And finally, s*ize* is used to indicate the logical distance relative to the collection(s) in use. Therefore, large sized circles indicate the collections presently in use (active or non-active); their immediate super- or sub-collections are of medium size; other super- and sub- collections are of small size.

The second issue is dealt with by the *Content area* (B), which is the place where application windows, the documents contained in a collection (Figure 1 shows a temporal overview of a collection), etc. are displayed. Hence, if the user is for instance working on some text file, the file is displayed in this area and may be enlarged to cover the whole screen real estate if the user wishes so

The third issue is dealt with by the *Context/Attribute area* (C). The top section (*current context)* indicates the active context for the active document or collection. The central section (*belongs to*) displays other available (but non-active) contexts for the active document or collection. Through a menu bar, the remaining attributes (e.g. author, creation date) of either an active document or of an active collection can be accessed.

4.2   Dependencies

In order to work properly with dependencies between documents, these have to be visible from

two different perspectives: First, from an overview perspective that illustrates all categories and the occurring dependencies between them; and second, from the perspective of an individual document itself.

The overview is provided by *named arrowed lines* in the Overview area (A) that indicate crosslinks (associations) between selected members of the participating circles (collections). An active dependency is indicated by a bold arrow.

For an individual document, finally, its links to or from other documents in the system are explicitly listed (*linked with*) in the Context/Attribute area (C).

5   Conclusion and Outlook

In the task walk-throughs, our users interestingly did not seem to need the entire power we provided. It seems as if a carefully evaluated, reduced set of functionality may be enough to serve the needs that a users has in order to organise their information. This seems also the case with respect to selected details of the graphical representation. For instance, in our prototype, documents are linked by a *named* arrow. Whereas the role of the arrow was doubtlessly perceived as clear and useful, its naming was less so. With respect to the design itself, our users positively valued our decision to use unobtrusive, 'boring' colours instead of a bright, colourful design. The graph as an abstract metaphor was valued basic, but suitable and positively non-distracting from the task at hand. As a consequence, additional work is still required to refine the details of the graph as representational means on the one hand, and the access procedure to system owned functionality on the other.

Our users were of the opinion that we had managed to provide them a hands-on experience with the system. Thanks to the careful definition and afterwards modelling of the typical working tasks in the PowerPoint prototype, they were never at a lack of 'real' functionality. Their reactions to the prototype suggest that they are intuitively familiar with all of the paradigms mentioned in Section 1. This is of particular interest in the case of the three users entirely new to our ideas. We did not need to explain the functionalities explicitly to them, but the mere mention of their existence was enough to capacitate our users to use them for the accomplishment of the tasks. Their feedback throughout was positive. The availability of multiple classification and bi-directionally linked documents, for instance, was warmly welcomed.

In the future, three aspects will be in the focus of our efforts: (1) The UIP also integrates support for multiple users and distribution. These aspects have so far not been considered at all in the existing prototypes. (2) We still consider it very much worth advancing the remaining two cardboard and paper prototypes. We expect different user feedback that enlightens the same concepts from a different angle. (3) The proposed prototypes will naturally also be implemented into fully functional interfaces with subsequent further user studies in order to keep them appropriate.

All in all, we think that our undertaking has been successful: Our meticulous procedure implied a large amount of effort, but enabled us to obtain fundamental user feedback - indispensable for a project such as ours. For the first time, users were able to judge *the whole set* of novel paradigms by a hands-on experience and their reactions encourage us to continue on the chosen track.

References

Dourish, P., Edwards, W. K., LaMarca, A., Lamping, J., Petersen, K., Salisbury, M., Terry, D. B., & Thornton, J. (2000). Extending Document Management Systems with User-Specific Active Properties. *ACM Transaction on Information Systems, 18*(2), 140-170.

Dourish, P., Edwards, W. K., LaMarca, A., & Salisbury, M. (1999). Presto: An Experimental Architecture for Fluid Interactive Document Space. *ACM Transaction on Computer-Human Interaction, 6*(2), 133-161.

Fertig, S., Freeman, E., & Gelernter, D. (1996). *Lifestreams: An Alternative to the Desktop Metaphor.* Proceedings of the ACM Conference on Computer-Human Interaction (CHI) 1996.

Freeman, E., & Fertig, S. (1995). *Lifestreams: Organizing your Electronic Life.* Proceedings of the AAAI Fall Symposium: AI Applications in Knowledge and Retrieval, Cambridge, MA.

Norrie, M. C., Steiner, A., Wuergler, A., & Wunderlin, M. (1996). *A Model for Classification Structures with Evolution Control.* Proceedings of the 15th International Conference on Conceptual Modelling (ER'96), Cottbus, Germany.

Rekimoto, J. (1999). *TimeScape: A Time Machine for the Desktop Environment.* Proceedings of the ACM Conference on Computer-Human Interaction (CHI) 1999.

Rivera, G., & Norrie, M. C. (2002). *A Database Approach to Global Document Spaces: Replacing Files with Shared Connected Objects.* Proceedings of the Cooperative Information Systems Conference (COOPIS 2002), Irvine, CA.