Understanding Stories about Personal Documents

Daniel Gonçalves, Tiago Guerreiro, and Joaquim A. Jorge
Department of Information Systems and Computer Engineering
Instituto Superior Técnico / Technical University of Lisbon
Av. Rovisco Pais, 1000 Lisboa, Portugal
daniel.goncalves@inesc-id.pt, tjvg@immi.inesc-id.pt, jaj@inesc-id.pt

Abstract

Hierarchies are the most common way to help users organize their personal information. However, their use is fraught with problems. In particular, the users' documents, stored in the filesystem, are notoriously difficult to manage and retrieve. A way to alleviate those problems is to resort to a wide range of knowledge about the users, their actions and the world that surrounds them. Systems that have done so lack an organizing principle that helps users refer to all the information that might be relevant. We propose a novel user interface paradigm, narrative-based interfaces, that provides such an organizing principle, showing how knowledge plays a central role in understanding the users' stories. An analysis of Quill, a prototype narrative-based interface for personal document retrieval, will show how narratives can successfully be used to help users retrieve their documents.

1. Introduction

The use of hierarchies as a way to help users organize their personal information is widespread. However, classifying all items into the hierarchy is not an easy task. While until recently the number of documents each user had to deal with was sufficiently low to be manageable using hierarchic classifications, this is ceasing to be the case. Electronic documents now pervade our daily lives, in their different forms. From letters and other text documents to digital photographs or videos, both the numbers and types of personal documents have suffered a hitherto unseen growth in recent years. Despite this fact, little has been done to help users manage them in novel and meaningful ways.

Recently, desktop search systems, such as Google Desktop, have become popular. However, the interactions they provide are fairly restricted, due to the limited expressive power they possess. Borrowing from the paradigm that has pervaded Internet search since its inception, keyword search, most of today's desktop search programs make it impossible to use information other than keywords that might appear in documents to retrieve them. Left out is a wealth of relevant information about the users, their documents, and the context in which they were handled, much of which might be more meaningful to users than arbitrary classifications in hierarchies. Indeed, studies of the users' email inboxes [14] have shown that email tools are often used to manage documents. Every message in a users' inbox is associated to information such as its sender and subject, which make it possible for users to organize and retrieve their documents more easily than dealing with the filesystem, even considering that email tools provide no explicit support for those tasks.

Recognizing the usefulness of autobiographic information about documents, some studies have tried to use it to help users manage those documents. In Placeless Documents, documents could be organized in "virtual collections" defined by filters of the document's metadata [3], and automatically updated whenever new relevant documents appear. This idea was also used by Baeza-Yate's PACO [2]. More recently, in Stuff-I've-Seen [4], all information elements handled by the user are indexed, and can then be retrieved with the help of keyword-based search, after which the results can be filtered using the available meta-data. MyLifeBits [5] aims at being able to automatically record all information relevant for any given user (contacts, documents, email messages, events, photos, music, video, etc.), each with its own meta-data properties. Items can be linked together if they are somehow related (a photo to the contacts of the persons shown in it, for instance). The resulting interrelations graph can help users navigate their "bits"

of information in search of a specific one. Soule's Connections search tool [12] monitors file system calls and creates a graph of related documents based on when they were handled. The users can then find their documents by navigating the graph. Kim's Personal Chronicling Tools [9] monitor the opening and modifying of documents, placing content into the clipboard, sentences entered using the keyboard, applications used, instant messages sent and received, etc. As is the case for Stuff-I've-Seen, search results can be filtered based on the available meta-data.

All the above systems handle autobiographic information in limited ways. Meta-data usually plays a secondary role, allowing the filtering of results only after a keyword search has been performed. Also, the systems' interfaces do not make it easier for users to recall relevant autobiographic information. Filling in values for arbitrary properties might be as cumbersome as resorting to hierarchies.

We have developed a new interaction paradigm, narrative-based interfaces, in which stories about the users' documents can be told, providing enough information to the computer to find those documents. The information elements in stories appear not as unconnected data tidbits, but as a coherent whole. As such, they will capitalize on the human's associative memories to help them recall relevant information. Furthermore, narrative-based interfaces are natural, as humans are natural-born storytellers.

In the next section, we will describe how our prototype narrative-based interface, Quill was designed, after which the interface itself will be described. A discussion of how stories can be understood and used by the interface will ensue. Then, we will provide some experimental results that show how the information contained in stories can be, with the help of the common-sense knowledge, used to understand them and successfully retrieve personal documents. We will then conclude, pointing to possible future work.

2. Studying Stories About Documents

To understand what to expect from document-describing stories, we interviewed 20 users, collecting 60 such narratives. Resorting to contents and relational analysis [8], we identified the different kinds of story elements that might appear in stories, their relatively frequencies, and the expected transition probabilities between those elements [7]. This made it possible for us to infer, using Hidden Markov Models, archetypical

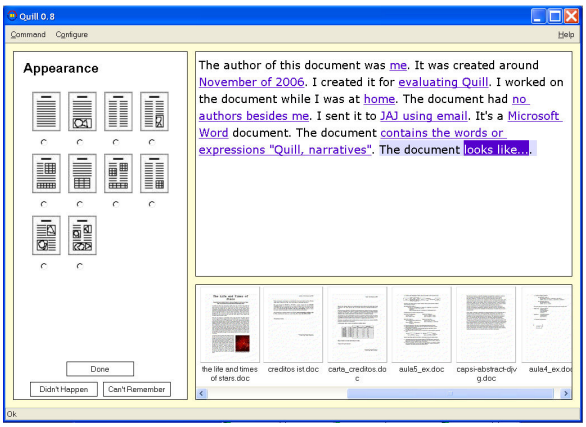

Figure 1. The Quill narrative-based interface

story structures to be used in Quill to guide the storytelling process and allow the system to build expectations to better understand it the stories.

The qualitative analysis of the story transcripts also yielded relevant results. Most notably, we found it is important to maintain dialogues with the users, to prevent them from digressing and also to help jog their memories and recall more relevant information. Also, it was verified that knowledge about the user and the world is essential to understand the stories, as much of the information required to comprehend them is taken for granted by the users.

After gaining a thorough insight of document-describing stories, two low fidelity prototypes of possible interfaces were created and evaluated by users [6]. The one that allowed stories more similar to those told to humans to be told was chosen. It better maintained the illusion of storytelling. Its development led to the creation of Quill, described below.

3. The Quill Interface

The Quill interface (Figure 1) allows users to tell their stories using a fill-in-the-blanks approach. Each possible story element is suggested in turn to the user. Whenever this happens, an incomplete sentence is appended to the end of the story. The missing information can then be entered with the help of a specialized dialogue to the left of the story area (one for each element). While the elements are suggested to the users in the archetypical order inferred from stories told to human listeners, they may choose another element at all times. The users can also mention that something didn't take place (the document had no co-authors, for instance), or that they do not remember something: not knowing something to have happened and knowing it didn't happen are two different things.

As the story is written, the system continuously searches for documents that match it. Thumbnails of those documents are presented to the user at the bottom of the screen. This takes advantage of the users' visual memories, allowing them to easily identify the target documents without disrupting the story flow.

4. Gathering Information

In order for Quill to use stories to find documents, it must access a wide range of knowledge. An index of the users' documents is necessary, as well as additional autobiographic information that can be used to understand the stories. To make this possible, we resort to Quill's Knowledge Base (KB). Relying on data explicitly provided by the users would undoubtedly fail. No one would be willing provide it. We prevented the need for such manual annotations by creating a plugin-based monitoring system that continuously observes what happens in the users' computers, selecting relevant information, and updating the KB.

4.1. Documents

The first time the system runs, all documents already in the users' machine are indexed. From then on, changes to those documents are continuously monitored. For each document, a wealth of information is stored in the KB, including all data that can be gleaned from the filesystem (filenames, creation dates, etc.). A more thorough processing of every document is also performed. Text-based documents are converted to plain text and tokenized. Then, the Porter stemming algorithm [10] is used to find the stems of the different words in the text. Finally, tfidf algorithm [11] selects the keywords that best represent the document. Also, all metadata associated to the documents is used (the ID3 tags of .mp3 and .ogg files, for instance).

4.2. email

By indexing email messages, the system knows what documents were sent or received by email, but also the subjects they were related to, the people the user knows, and when a document was handled or a subject considered.

As for documents, all emails already present in the users' machines the first time the system is ran are indexed and subjected to a treatment similar to the one for documents Two real-time plugins, that work as proxy POP3 and SMTP servers, keep the KB current.

All documents attached to email messages are also indexed as personal documents. If the document already exists somewhere in the filesystem, instead of creating KB entries for a new document, the email plugin simply annotates the existing document with the information that it was sent by email as an attachment.

4.3. Calendar

Also important to understand stories about are the users' datebooks, as they provide a glimpse of the wider context that surrounds them. All events are analyzed and stored in the KB.

4.4. Web

To understand the subjects the users were interested on, the news they were exposed to, and get a glimpse of what was happening in the real world while they handled their documents, all web pages visited by users and documents they downloaded are also inspected. As for the email plugins, some effort is taken in ensuring that no duplicate document entries are created.

4.5. Applications

This plugin continuously monitors the processes being run in the users' machines, and resorts to a list of relevant processes to filter those that might be relevant (Office applications, for instance), storing that information in the KB. Knowing which applications were used can help understand when tasks described mentioned by users took place, and when documents of certain types might have been handled.

4.6. Printer

By intercepting operating system events produced when a document is added to the print queue, it is possible to know when a document was printed.

5. The Knowledge Base

Our Knowledge Base uses RDF and RDF Schema as knowledge representation formalisms [13]0. We chose RDF based on a set of requisites inferred from the analysis of document-describing stories. The formalism had to be flexible, as the stories in which users describe their documents are varied and rich, and the knowledge

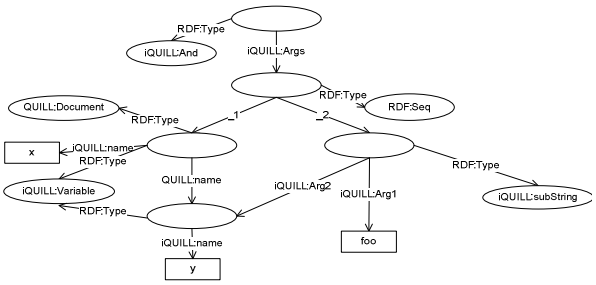

Figure 3. Sample iQuill inference rule. It returns as possible bindings for the x and y variables all documents whose names contain the string 'foo'.

required to understand them should be represented in an effective and uniform way. Also, not all elements are equally accurate. For instance, a reference to Time can be, for a document, that it was written "last Thursday after lunch", and for another "read last year around summer time". So, a single level of granularity cannot be imposed by the formalism. A pre-determined, non-extensible list of possible values is also out of the question. Finally, while the stories themselves convey lots of information about a user's documents, a large amount of knowledge is assumed to be known. It is the case of a document's Author, when referring to documents created by the users themselves. Things like a relative's birthday and the like are also taken for granted. In short, knowledge representing facts about the world and the user is required, and should be represented seamlessly with other autobiographic information. Also, whenever possible existing sources of knowledge should be reused.

RDF meets all these challenges. There is a continuum of increasingly expressive languages in the RDF family (RDF, RDFSchema, and three flavours of OWL) allowing us, if necessary, to upgrade the expressiveness of Quill at a later time, with little effort. It is a W3C standard that aims to bring semantic information to the World Wide Web. When this becomes a reality, such information will help us understand the users' actions when on- and off-line.

All knowledge in RDF is stored as a set of triples, in the shape `(<subject>, <predicate>, <object>)`. This is flexible, as it imposes little structure to what can be represented. However, if becomes cumbersome to deal with those triples directly. So, we created an abstraction layer on top of RDF which we called *Scroll*. Scroll allows more complex constructs than RDF triples to be handled with ease. The usual methods for KB interaction, Tell and Ask are available, but classes and their attributes can also be directly handled. Furthermore, Scroll was implemented as a semantic network. We implemented path and node-based inference, and designed a schema called *iQuill* (short for Quill Inference Package), that defines a series of case-frames for the representation of first-order logic (FOL) like formulae in RDF (Fig. 2). Scroll can use those case frames to perform inference. The expressivity of iQuill is similar to FOL, without the ability to represent negation and existential quantifiers (for computational efficiency reasons). On the other hand, procedural attachment and functions are allowed. It is possible, for instance, to compare two strings, or check the inequality of two numbers.

All knowledge gathered by the different monitoring plugins is stored in the KB using the *Quill* RDFS Schema. Two main classes are the basis for this schema. The `Document` class contains all fields required to store information about a document. The most straightforward of those fields allow the representation of data collected directly from the filesystem, such as a document's filename, size or extension. Other information, such as the document's creator, keywords or title can also be represented, as can references to different versions of the same document. Instead of creating subclasses for the different document kinds, we chose to store knowledge about all documents in an uniform way, using just the `Document` class. This prevents the need to treat some documents as special cases, and makes it easier for Quill to handle them.

The second major class, `Event`, allows the storage of every relevant action detected by the system: sending or receiving an email, accessing a web page, meeting a co-worker, etc. Each event has a start and finish time, a description, a set of participants and of documents involved in it. The Event class is used for every possible event, allowing them to be uniformly handled by Quill. Two fields, `eventType` and `eventDirection` allow the differentiation of several kinds of events, if needed.

If Jack sent an email to user Jill arranging for a meeting, with an attached figure, an `Event` of `eventType` email and `eventDirection OUT` would be recorded. Its participants would be Jack and Jill, it would point to a `Document` with information about the email body, and the figure as a related document.

Apart from the `Document` and `Event` classes mentioned above, two other auxiliary classes are defined in the Quill schema. The `Person` class represents a person, with multiple aliases and email

addresses. The `Locus` class is used to represent one of the users' machines, where documents might reside.

6. Understanding Stories

All the information in the KB is used by Quill to understand stories and retrieve documents. We will now describe how this is accomplished.

6.1. Natural Language Understanding

We constrained what the users can mention in their stories as much as possible, within reasonable limits based on the contents of stories gathered in the interviews. Even so, in some of the story element dialogues in Quill's interface, free-form text is allowed. Understanding that text becomes easier as the dialogue in which the text is entered provides the first clue to what its meaning might be. For instance, in the Time dialogue any text entered by the user is likely to describe an instant in time. The parsing of natural language (NL) sentences is, thus, performed by the different story element dialogues.

Sentences are first parsed using a chart parser and context-free grammars, specific grammar for each dialogue. We use augmented grammars to automatically derive the phrases' semantics during the parsing process. Assuming compositional semantics (the meaning of each component can be derived solely from those of its sub-components), each rule in the grammar is associated with a lambda calculus formula that is evaluated when the rule is applied to generate the semantics of the resulting phrase element.

This automatic extraction of semantics can be very helpful in some cases. It is the case of the dialogue for the Time element, which tries to parse text entered in it as a reference to a time instant. In this case, the semantics generated by the parser are timestamps that can then be directly compared with a document's creation or modification date.

If the chart parsing algorithm fails, a chunk parser [1] tries to decipher sentences one piece at a time. The entire sentence does not need to be correctly parsed, making the parser more tolerant to mistakes.

6.2. World Knowledge

Besides the knowledge gathered by the monitoring system, a different, more general kind of knowledge is also needed: common-sense knowledge about the world. For instance, if a user mentions a document was created around New Year's Eve, Quill must know that "New Year's Eve" is a holiday that occurs on January 1st of every year. This kind of knowledge was stored in the KB and used by the different story element dialogues whenever needed. Continuing the example above, if a chunk parser produced a noun phrase with the expression "New Year's Eve", it would look in the KB for some indication of what it could mean, and discover the date it refers to and its periodicity. In this way, instead of having to hard-code every such detail, Quill's understanding power can be enhanced just by providing it with more knowledge in the KB.

6.3. Searching for a Document

Whenever the user enters a new element into the story, a new set of inference rules is created by its corresponding dialogue. The different inference rules are then passed to the Document Searcher sub-module of Quill. It evaluates each of those rules in the KB. A score (positive or negative) would then be assigned to each document thus identified. The sum of all scores from all inference rules provides a ranking for all documents. Those better ranked (with higher scores) are suggested to the user in the document suggestion area of the interface as probable matches.

7. Evaluation

To find if narrative-based interfaces are actually able to help users retrieve their personal documents, and to what extent can the autobiographic information and knowledge stored in the knowledge base be of use, we performed a user study in which the document retrieval rate of Quill was measured. Twenty-one users were interviewed and asked to retrieve three different documents, for a total of 63 retrieval sessions. Before trying to retrieve the documents, Quill's monitoring system was allowed to index their emails, agendas, and documents. It is important to note that actual personal documents of the users were considered, instead of a pre-defined test-set. Only for their personal documents can users tell meaningful stories.

We found that, overall, Quill allowed he users to retrieve 87.9% of all documents sought. If considering only text-based documents, the value reaches 95.2%, whereas for non-text-based documents (photos, music, etc.) it drops to 68.8%. These are very good results for a general-purpose tool that does not have provisions for special cases such as photo properties. It is conceivable

that, with Quill running for an extended period of time on the users' machines, continuously gathering information, the results would improve, especially for non-textual documents, for which contextual autobiographic information plays an important role.

Another important result is that while keywords might have sufficed to find 64.7% of documents, information in the pathname would be required for 27.5% and, for 7.8% of documents, no textual information employed by the users would have helped find them: those documents were found solely with the help of other autobiographic information, understood based on the knowledge available to Quill.

8. Conclusions

Most tools used nowadays to help users manage their personal documents fail to employ a wide range of autobiographic information those users easily recall and associate to those documents. Keyword search is common and, even when other kinds of information can be used, they appear only as a way to filter the results of keyword search.

We've shown how narrative-based interfaces can help users convey relevant information about their documents to the computer, by telling stories about those documents. To understand those stories, it is important to possess a wide range of knowledge about the users themselves, the environment that surrounds them and their activities. We've seen how it is possible to automatically collect such knowledge that, together with common sense knowledge, can be employed to understand stories and retrieve documents. In fact, our prototype system, Quill, was able to successfully use it: 7.84% of all documents found using Quill would not have been found by approaches that make no use of additional knowledge.

In the future, it will be interesting to allow Quill to index other relevant information sources, such as instant messaging exchanges and SMS. As ubiquitous computing becomes more of a reality, it will become possible to gather a richer set of autobiographic information, referring to the users' activities away from the computer. Also, we hope to see how narratives can be used for other domains, such as structured document annotation or the retrieval of real-world objects.

9. Acknowledgements

This work was supported in part by project BIRD, FCT POSI/EIA/59022/2004

10. References

[1] S. P. Abney. Parsing by chunks. In S. P. Robert, C. Berwick and C. Tenny, editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages 257–278. Kluwer Academic Publishers, 1991.

[2] R. Baeza-Yates, T. Jones, and G. Rawlins. A New Data Model: Persistent Attribute-Centric Objects, *Technical Report*, University of Chile, 1996.

[3] P. Dourish, W. K. Edwards, A. LaMarca, J. Lamping, K. Petersen, M. Salisbury, D. B. Terry and J. Thornton. Extending Document Management Systems with User-Specific Active Properties. *ACM Trans. on Information Systems, 18(2)*, pp140-170, ACM Press 2000.

[4] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin and D. C. Robbins. Stuff I've Seen: a system for personal information retrieval and re-use. *In SIGIR '03: Proceedings of the 26th annual international ACMSIGIR conference on Research and development in informaion retrieval*, pages 72–79. ACM Press, New York, NY, USA, 2003. ISBN 1-58113-646-3.

[5] J. Gemmel, G. Bell, and R. Lueder. MyLifeBits: a personal database for everything. *Communications of the ACM*, 49(1), pp. 88-95, 2006.

[6] D. Gonçalves, and J. Jorge, Telling Stories to Computers. *In Proceedings CHI2004*, ACM Press, 27-29 April 2004, Vienna, Austria.

[7] D. Gonçalves, and J. Jorge. "Tell Me a Story": Issues on the Design of Document Retrieval Systems. In *Proceedings DSV-IS'04*, Lecture Notes on Computer Science, Springer-Verlag, July 2004, Hamburg, Germany.

[8] M. Huberman, and M. Miles. *Analyse des données qualitatives. Recueil de nouvelles méthodes.* Bruxelles, De Boeck, 1991.

[9] P. Kim, M. Podlaseck and G. Pingali. Personal chronicling tools for enhancing information archival and collaboration in enterprises. *In CARPE'04: Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences*, pp .56–65. ACM Press, New York, NY, USA, 2004. ISBN1-58113-932-2.

[10] M. F. Porter. An algorithm for suffix stripping. *Program* 14, pages 130-137. 1980.

[11] G. Salton. *Automatic Text Processing*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1988.

[12] Craig A. N. Soules and Gregory R. Ganger. Connections: using context to enhance file search. In SOSP '05: Proceedings of the twentieth ACM symposium on Operating systems principles, pages 119–132. ACM Press, New York, NY, USA, 2005.ISBN 1-59593-079-5.

[13] W3C Semantic Web, http://www.w3.org/2001/sw/

[14] S. Whittaker, C. Sidner. Email overload exploring personal information management of email. In *Conference proceedings on Human factors in computing systems*, pages 276-283, ACM Press, 1996.