Authoring and annotation of desktop files in *seMouse*

Oscar Díaz, Jon Iturrioz, Sergio F. Anzuola
{oscar.diaz, jon.iturrioz, jibfeans}@ehu.es

The Onekin Group - University of the Basque Country
P. O. Box 649, P^{o}. Manuel de Lardizabal, 1, 20.018 San Sebastián (Spain)

Abstract. Coping with an increasing number of files is one of the challenges of current desktops. Adding semantic capabilities is one possible solution. Aligned with this proposal, this work introduces the notion of "knowledge folder" as a coarse set of documents bound together by a common ontology. The ontology plays the role of a clipboard which can be transparently accessed by the file editors to either export (i.e. annotation) or import (i.e. authoring) metadata within the knowledge folder. Traditional desktop operations are now re-interpreted and framed by this ontology: copy&paste becomes annotation&authoring, and folder digging becomes property traversal. However, a desktop setting requires seamless tooling for these ideas to get through. To this end, this work proposes the use of the mouse as the "semantic device". Through the mouse, the user can classify, annotate, author, and locate a file as a resource of the underlying ontology. Moreover, being editor-independent, the mouse accounts for portability and maintainability to face the myriad of formats and editors which characterizes current desktops. The "semantic mouse" is implemented as a plug-in for *Windows*.

1 Introduction

Current desktops should be enhanced with mechanisms that permit users to abstract away from files.This work builds on the notion of **knowledge folder**, i.e a coarse set of information elements bound together by a common ontology. The folder contains an ontology, the instantiations, and the resources being annotated. A desktop can hold distinct knowledge folders, and a given file can belong to several knowledge folders. In contrast with current folders, this mechanism attempts to abstract away from how files are physically organised, by providing an ontology-based organisation.

As an example of a knowledge folder, consider all the documentation that goes with a research project. This includes the project proposal (e.g. a Word file), bills being payed by the project funds (e.g. Excel files), etc. These files can be scattered around distinct (physical) folders. Even though, they can belong to the same knowledge folder as some of the following clues indicate,

- data replication among documents (e.g. the funding body appears in the proposal but it is also acknowledged on the articles),
- simultaneous access. More than a document is accessed during "a typing session" (e.g. when writing the article, the proposal is checked out for the submission deadline),

– event correlation. Creation/removal of the documents are not totally independent (e.g. a bill does not exist without a project proposal), etc.

Current desktops ignore this situation and treat files as isolated units. Today, we copy and paste the text values from one file to another, and the ontology is kept (and managed) in the users mind. And too often, file location turns into digging through a hierarchical folder tree.

This paper presents how this situation can be improved by the introduction of knowledge folders. Specifically, copying&pasting becomes annotating&authoring, and folder digging becomes ontology traversal. By tapping current file structures into an ontology, authors can both populate the ontology (i.e. annotation), and reuse the instances of the ontology while authoring a document. The ontology instances play the role of a clipboard which can be transparently accessed by the file editors to either export (i.e. annotation) or import (i.e. authoring) metadata within the knowledge folder. As for file location, files are now resources of an ontology. This permits to enhance and contextualize desktop search based on the ontology properties, and navigate along the ontology associations.

Being in a desktop setting, we can not ignore usability. Handling of knowledge folders should be as seamless as possible. Rather than providing separate tools for exporting/importing (i.e. annotation/authoring), we strive to accommodate to the current tools for traditional copy&paste operations: the mouse. This will certainly facilitate user adoption.

To this end, the *semantic mouse* (*seMouse*) is introduced. By clicking on its middle button, *seMouse* exports/imports properties from the *ontology,* regardless of the editor you are working with. It does not matter whether you are working with *Word, PowerPoint, Netscape*, etc, the "semantic" button is available for annotation/authoring. In this way, the user does not have to move to a new editor when annotating (like in *SMORE* [2]), nor has to learn a new "ontological interface" when files from different formats are edited (like in *SemanticWord [3]*).

Both, the support of knowledge folders as the underlying infrastructure, and the use of the mouse as the device to interact with this infrastructure, are the main contributions of this work towards making desktops semantic.

Next section introduces *seMouse* through five scenarios, namely, file classification, annotation, authoring, semantic navigation and ontology editing.

2 *seMouse* at work

seMouse is an annotation/authoring device that achieves editor-independence by working at the operating-system level: the mouse. This section introduces *seMouse* with the help of an example.

As a knowledge folder, consider the cluster of heterogeneous documents that goes with a research project. This includes the project proposal (e.g. one Word file), bills payed with the project funding (e.g. twenty *Excel* files), papers as deliverables of the project (e.g. twenty files in both *.pdf* and *.doc* formats), participants (whose desktop counterpart can be either the homepage, an *.html* resource, or a *.pdf* resource) and comments (being realized as either emails or .doc resources).

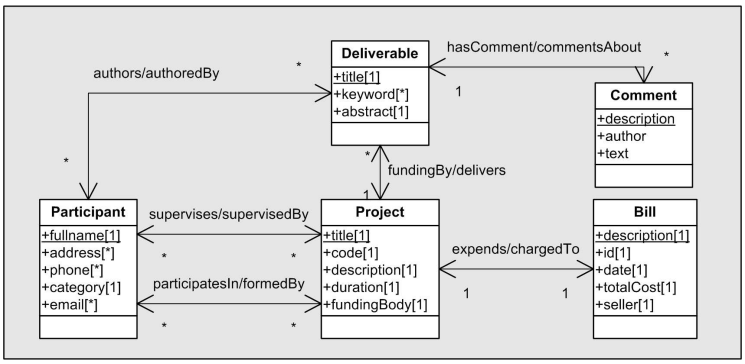

Fig. 1. A sample ontology.

Regardless of their format and folder location, it is likely that a high degree of content reuse as well as frequent contextual navigations within this "file space" happens. *This is what makes this set of files a knowledge unit.* Being in a participant -an *html* file-, you frequently need to locate her project proposals -Word files-, or being in a project proposal, the associated papers -*PDF* files- are commonly accessed.

A knowledge folder comprises an ontology (figure 1 shows the one for the sample problem). Five classes are identified. Each class is characterized by a set of value-based properties (e.g. *title, keyword, abstract*). Associations are defined between these classes (e.g. a project is *supervisedBy* a participant) (termed *ObjectProperty* in OWL). And the expressiveness of OWL can be used to define inverse and transitivity properties between the associations.

Although the ontology is at the core of the semantic desktop, this paper focuses on authoring and annotating resources of the ontology. We do not address how the inference power of the ontology can achieve its full potential in a desktop setting.

Once the ontology has been set, the population process begins. The key idea is to use the mouse as the semantic device so that interactions with the underlying ontology are achieved via mouse clicks. Specifically, pressing the middle button on the mouse causes an interaction with the ontology manager. This interaction is context-sensitive, i.e. the button accomplishes distinct operations depending on the place the pointer sits on. Next paragraphs introduce five scenarios of the use of the semantic mouse.

Scenario 1: file classification. First of all, files need to be identified as instances of any of the ontology's classes. This is achieved by opening a file, and pressing the middle button. A menu pops up for the user to indicate to which class this file is a resource.

Scenario 2: annotation (see figure 2 and 3). Annotation&authoring becomes the counterpart of copy&paste in traditional desktops, with the difference that now these operations are conducted along the ontology net. What is being exported(i.e. copy) is no longer a string but a class property of the ontology.

If a file has already been categorised, the annotation process may begin. If some text is selected, the mouse is used to export this text as part of the value of a property as it is shown in figure 2. Of course, the set of properties will depend on the class of

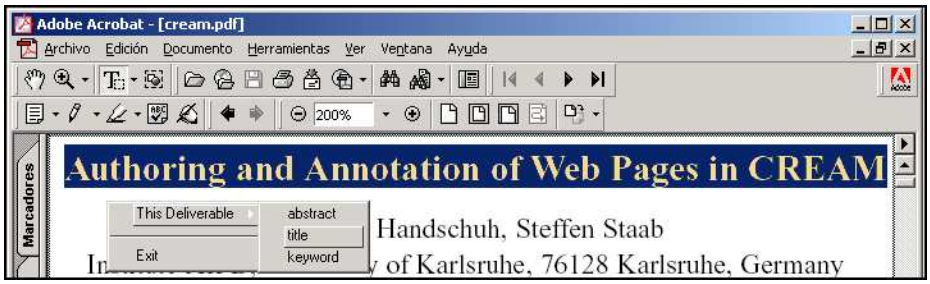

Fig. 2. Scenario 2: annotation. Some text is selected. Being a *deliverable* file, the menu displays properties of this class. The text will become the value of the chosen property.

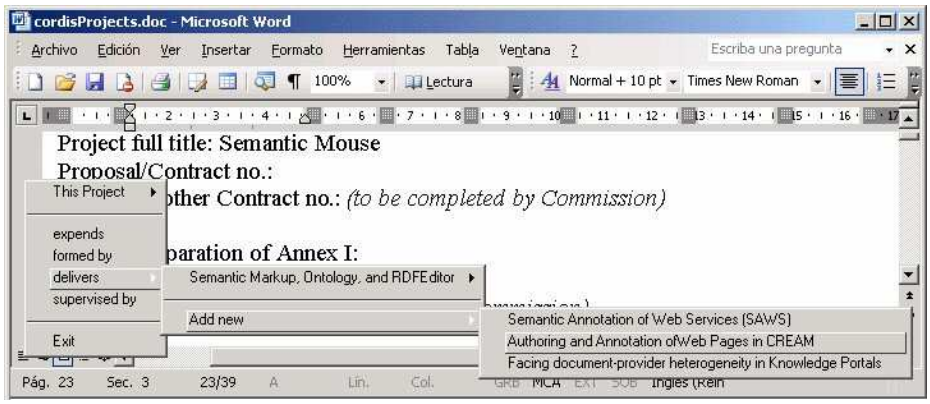

Fig. 3. Scenario 2: annotation. No text is selected. The menu shows associations of the file class.

the resource. In the example, *title, keyword* and *abstract* correspond to properties of the *deliverable* class.

On the other hand, if no text is selected, the middle button is used to establish associations with other files. This situation is exemplified in figure 3. In this case, the CORDIS project template for EEC projects has been used. When the middle button is pressed, a menu pops up for the user to link the current resource with other target resources. The menu is customised for the current resource, that is, the associations are restricted to those available for the current resource, whereas the target files are also limited to those of the appropriated class. In the example, the association can only be established with *deliverables* files since this is the destination class of the *delivers* association.

Scenario 3: authoring (see figure 4). Associations being set during annotation can now be exploited. For instance, the *project* resource can import the title of its associated *deliverable* resources. In the example, the article *"Authoring and Annotation of Web Pages in CREAM"* appears as a *deliverable* of the current file. By selecting this article,

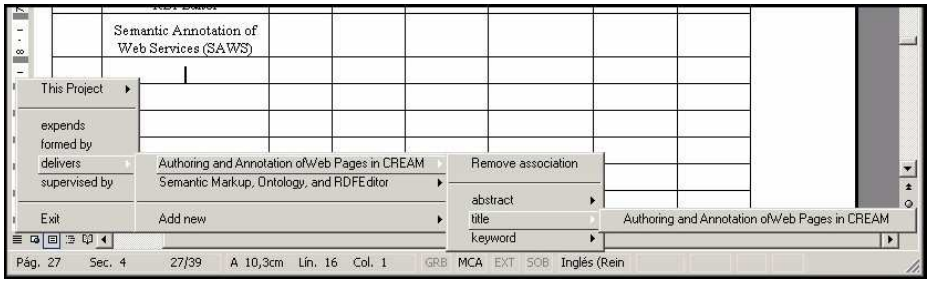

Fig. 4. Scenario 3: authoring. The *title* of a *deliverable* is imported into a *project* resource.

the menu is extended rightwise to show up its properties. The user can select one of these properties, and its value is inserted at the cursor place.

Scenario 4: semantic navigation . File location in current desktops frequently implies folder digging. By contrast, semantic navigation strives to exploit the associative behaviour of the human memory. A resource can be located from the resources it is related to. That is, the ontology provides the context to facilitate resource location.

Once a file has been selected, semantically-related files can be located by pressing the middle button, regardless of the folders where these files are physically located, providing a resource-centric navigation. This facilitates location of neighbour resources, but it may be cumbersome whenever browsing is required. In this case, a graph-based RDF visualizer can be a better option (see [1] for an overview of RDF visualizers).

3 Conclusions

This work strives to lower the adoption barrier of the semantic desktop by providing seamless tooling. To this end, we support the notion of "knowledge folder" as the underlying infrastructure, and the "semantic mouse" as the interactive device. Being editor-independent, the mouse accounts for portability and maintainability to face the myriad of formats and editors which characterizes current desktops. Similar to other areas of computing, a balance is needed between generality (e.g. format-independence, editor-independence, etc), and functionality (i.e. the semantic tooling available). *seMouse* illustrates a semantic-lite approach where a compact set of functions are available to no matter which editor within *Windows*.

References

1. John Gilbert and Mark H. Butler. Review of existing tools for working with schemas, metadata, and thesauri. Technical report, Hewlett Packard Laboratories, October 2003.
2. Aditya Kalyanpur, James Hendler, Bijan Parsia, and Jennifer Golbeck. SMORE - Semantic Markup, Ontology, and RDF Editor. http://www.mindswap.org/papers/SMORE.pdf, 2004.
3. Marcelo Tallis. Semantic Word Processing for Content Authors. In *Workshop Notes of Knowledge Markup and Semantic Annotation Workshop (SEMANNOT 2003). Second International Conference on Knowledge Capture (K-CAP 2003)*, October 2003.