Capture of lifecycle information in office applications

Lasse Lehmann*, Arno Mittelbach,
Christoph Rensing and Ralf Steinmetz

KOM – Multimedia Communications Lab,
Technische Universität Darmstadt,
Merckstrasse 25, 64283 Darmstadt, Germany
E-mail: Lasse.Lehmann@kom.tu-darmstadt.de
E-mail: Arno.Mittelbach@kom.tu-darmstadt.de
E-mail: Christoph.Rensing@kom.tu-darmstadt.de
E-mail: Ralf.Steinmetz@kom.tu-darmstadt.de
*Corresponding author

Abstract: Due to the continuously growing number of Learning Resources the automatic generation of metadata has become an important research issue. Lifecycle information like, relations resulting from reuse of Learning Resources can be utilised to support users, for example, in retrieving relevant documents. However, this kind of information is neglected by most systems. We show that lifecycle information can be obtained automatically without user interaction. We describe the challenges that have to be faced when capturing lifecycle information in office applications and present how we met them in the implementation of add-ins for our LIS.KOM framework. The results of first evaluations are promising.

Keywords: lifecycle information; metadata; relation information; document lifecycle; office documents.

Reference to this paper should be made as follows: Lehmann, L., Mittelbach, A., Rensing, C. and Steinmetz, R. (2010) 'Capture of lifecycle information in office applications', *Int. J. Technology Enhanced Learning*, Vol. 2, Nos. 1/2, pp.41–57.

Biographical notes: Lasse Lehmann is pursuing his PhD at the Multimedia Communications Lab at Technische Universität Darmstadt. In 2005 he graduated as Dipl-Ing in Electrical Engineering and Information Technology from Technische Universität Darmstadt. His research interest lies in the area of knowledge media. He is especially interested in the capture and utilisation of information emerging throughout the lifecycle of documents.

Arno Mittelbach received his BSc Thesis on "Capture and Verification of Lifecycle Information for Text-Based Documents" at the Multimedia Communications Lab at Technische Universität Darmstadt in 2008. He then worked as a research programmer for Oxford University Computing Services before embarking on his MSc Programme in Computer Science at Technische Universität Darmstadt in April 2009.

Christoph Rensing completed his PhD thesis in 2003 on "A policy-based access control architecture for the Multi-Service Internet". He works as Research Group Head at the Multimedia Communications Lab at Technische Universität Darmstadt, leading a team of researchers in the area of Knowledge Media. In addition, he is a research consultant at the Hessian Telemedia Technology

Competence Center. He has published more than 60 papers in national and international conferences and journals. He is a member of the ACM, and GI.

Ralf Steinmetz worked for over nine years in industrial research and development of distributed multimedia systems and applications. He has been Head, since 1996, of the Multimedia Communications lab at Technische Universität Darmstadt, Germany. From 1997 to 2001 he directed the Fraunhofer (former GMD) Integrated Publishing Systems Institute IPSI in Darmstadt. In 1999 he founded the Hessian Telemedia Technology Competence Center (httc e.V.). His thematic focus in research and teaching is on multi-media communications with his vision of real "seamless multimedia communications". With over 200 refereed publications he has become ICCC Governor in 1999; was awarded the ranking of fellow of both, the IEEE in 1999 and the ACM in 2002.

1 Introduction

The automatic acquisition of metadata for Learning Resources, as well as for documents in general, has been a research issue for quite some time (Duval and Hodgins, 2003) and several efforts have been made to generate metadata automatically (e.g., Meire et al., 2007). The number of documents stored on local computers is steadily growing and users tend to have problems organising their documents (Nejdl and Paiu, 2005). Additional metadata can be used to overcome problems like this.

When creating Learning Resources, authors usually re-use or re-purpose existing documents or parts of them. This specifically applies to Learning Resources created with office applications like Microsoft PowerPoint or Word (Klerkx et al., 2006). When, for instance, creating slides, many authors start with a search for existing presentations created by themselves or by colleagues. Processes like re-using, merging, adapting and re-creating Learning Resources provide for the emergence of a multitude of information (see Rensing et al. (2005) for our definition of Learning Resources as well as details on re-purposing processes). This includes relations that connect Learning Resources, documents and media objects which have been involved in re-use processes. For instance, when an author takes slides from one presentation to create a new one, a relation between the two presentations is created. The same applies when text or other content is copied to a Word document from another document or website. This relation information can be very helpful for the retrieval of Learning Resources, e.g., by providing links to related resources or as input for ranking and recommendation methods. Additionally, it can be used to support authors of Learning Resources, e.g., by notifying them when related resources are changed. Due to its nature, the information is bound to specific processes, i.e., the information can only partially, or not at all, be obtained in the aftermath. In a community scenario where relation information is collected from different authors networks of interconnections and relations between documents will be the result that can be visualised, browsed or searched. However, most existing systems do not capture this information and thus it is lost. We propose that this so called lifecycle information should be captured.

In order to achieve this, corresponding processes, i.e., the users' actions in respective applications have to be monitored. This is not always an easy task since, for most

widespread applications, the underlying source code is not available and even if it was available creating customised versions is, for various reasons, not desirable We designed and implemented a framework for the capture, management and utilisation of lifecycle information (see Lehmann et al., 2008a). In this paper we identify and resolve the challenges of lifecycle information capturing in office applications: Microsoft PowerPoint and Microsoft Word. This is done by means of ReCap.KOM add-ins for our LIS.KOM Framework. In Section 2 we introduce lifecycle information and show what types of lifecycle information we capture in PowerPoint and Word and how it can be utilised. In Section 3 we examine related work. After the description of the challenges we faced (Section 4) we provide details on our LIS.KOM Framework and show how we resolved the challenges in our implementations for PowerPoint and Word (Section 5). In Section 6 we present the results of a first evaluation. Section 7 concludes and gives an outlook on the steps we plan to do in future.

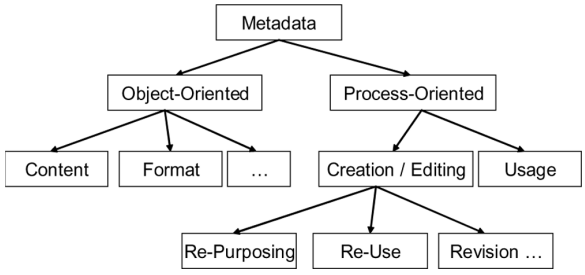
2 Lifecycle information

In the following we define lifecycle information, how it relates to metadata and the different types of lifecycle information we distinguish.

2.1 Definition of lifecycle information

For us, lifecycle information is a special type of metadata. In contrast to the common notion of metadata it is not bound to a specific object (object-oriented) but emerges from a specific process. Thus, lifecycle information is a synonym for process-oriented metadata (see Figure 1). The name 'lifecycle information' was chosen to reflect the fact that processes providing for the emergence of this information may occur during the whole lifecycle of a document. Another important feature of lifecycle information is that it is collected document centric; i.e., that the information captured is not bound to a specific user, system or application, but only to the related documents. We distinguish two general types of lifecycle information: context information and relation information.

Figure 1 Schematic metadata taxonomy



Source: Lehmann et al. (2008a)

Context information depicts the different contexts that a document went through during its lifecycle. This includes information about applications used to edit, store or present documents, durations and timestamps, persons that interacted with a document, numbers of accesses, queries and many more. Dependent on the applications and processes the

amount of context information that can be captured during the whole lifecycle of a document can be very big. There is already a considerable amount of related work dealing with this kind of lifecycle information (see Section 3).

The other type of lifecycle information we consider is relation information. A relation always connects two document instances in a specific way. There are different relation types emerging from different kinds of processes. However, the most common type of relation in our scenario originates from re-use processes.

As stated before, there are already some approaches dealing with the capture and utilisation of context information; however, relation information is neglected by most existing approaches.

Therefore, we decided to concentrate on the capture of relation information and consider context information only in connection with relation information in those cases where it adds to the meaning or information value of a relation. For instance, the weight of a re-use relation correlates to the author of the re-used document. We provide further details on lifecycle information, including a basic lifecycle information scheme in Lehmann et al. (2008a).

2.2 Lifecycle information in office applications

In the following section we present the different kinds of lifecycle information that we capture in our considered use case – office applications. Documents created with these applications naturally have a high potential for re-use. Presentations for instance, are rarely created from scratch. A first evaluation that we conducted has shown that presentations are usually created based on existing content. For text based documents the same applies. Often, text from existing documents, especially text from websites, is re-used when new documents are created. Therefore, re-use relations make up the lion's share of the lifecycle information we capture in office applications. Libbrecht (2008) model re-uses and identifies five re-use types. Three of these types are common practice during the creation of documents:

1 *Verbatim inclusion*: Including content without changing it. This is often applied to media objects like pictures or clip-arts.

2 *Copy-and-paste*: Content is copied from one document to another in order to be re-used. A common practice for PowerPoint as well as for Word documents.

3 *Copy and branch of large bodies*: Take a whole document, adapt, edit or rearrange it and save it as a new instance.

Although there are differences between Word and PowerPoint regarding the content being re-used, the actual information types that we capture are roughly the same. Lifecycle information that we capture includes:

- Relations resulting from application internal re-use (slides, text or other content).
- Relations resulting from the re-use of content from external documents (either between PowerPoint and Word or from completely external applications, e.g., PDF viewers, internet browsers, etc.).

- Relations resulting from re-use of media objects (like e.g., pictures, audio or video files) in Word or PowerPoint. This includes aggregation relations connecting the document with the media objects themselves (e.g., by 'isPartOf') and 'secondary relations' between two documents re-using the same object.
- Variant (or identity) relations connecting different document instances (e.g., when a document is saved under a different name or copied within the file system).
- Additionally, we capture context information (like user, author, computer information etc. and timestamps).

To construct the relations named above we basically use three types of re-use relations. For re-use of content between two documents sharing the same aggregation level – usually Libbrecht's second re-use type (Copy-and-Paste) – we use *providesElementTo/containsElementOf* relations. Although the relation itself resides on the document level, we additionally store information about the elements inside the documents that took part in the re-use process; e.g., the slide or paragraph that was copy-pasted. Thus, the relation actually points into the document and not only at the document.

To represent re-use of media objects from the file system (re-use type one), we use an aggregation relation (*isPartOf/hasPart*). Finally, for the third re-use type (Copy and Branch), we use *isVariantOf/hasVariant* to connect the documents involved in the process (see Lehmann et al. (2008a) for more details on relation types). The relations are stored in a structured manner following the scheme we described in Lehmann et al. (2008a). Each relation consists of the IDs of the source and target documents, a type, the creator field naming the person who created the relation and a timestamp. In case of *isPartOf/hasPart* or *providesElementTo/containsElementOf* relations we store additional information such as an element ID, element type, a hash value and for textual relations the content that has been re-used. The aforementioned theme has been extended to additionally hold asymmetric similarity values for each relation.

2.3 Utilisation of lifecycle information

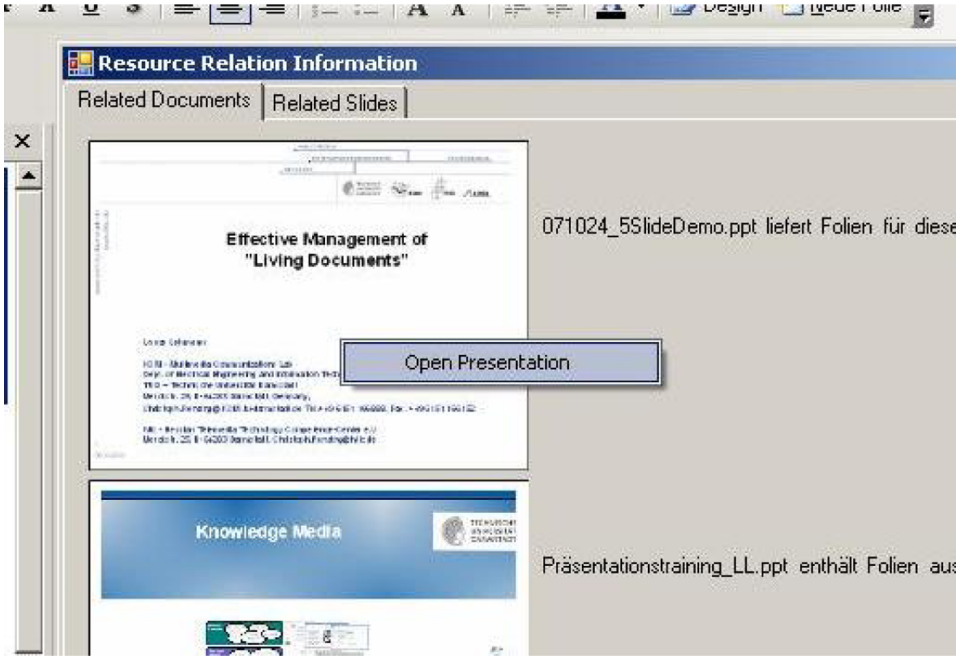
There are multiple possibilities to utilise lifecycle information, specifically relation information. Since this paper is mainly about the capturing of said information this section outlines only some examples.

A simple type of utilisation application that we have implemented as add-ins for both Word and PowerPoint provide the possibility for the author to view the relations the current document possesses. It shows previews of related documents and authors can open them directly, if stored on their computer, or open them from the LIS.KOM server, if the selected document is a foreign document. This option is specifically interesting when a different author re-used another one's content. Figure 2 shows a screenshot of the PowerPoint add-in.

Another way to utilise relation information that we plan to implement is a notification system. Authors can choose to be notified if documents containing content they re-used are updated or if their content is re-used in other documents. Here we consider not only direct relationships but also propagate the information along the relation branches. Typically, specific variant relations tend to build up long chains of similar documents, depending on the authors' working behaviour. Thus, the *isPartOf/hasPart* and

providesElementTo/containsElemtentOf relations along the way should also be considered.

Figure 2 Utilisation add-in in PowerPoint (see online version for colours)



Source: Lehmann et al. (2008a)

A further possibility is a stand-alone application. This could be a search tool or explorer, which visualises relations between documents and uses them to recommend documents or rank search results.

3 Related work

In this section we analyse work that is closely related to the approach presented in this paper. For a more detailed analysis of related approaches in general we refer to Lehmann et al. (2008a).

McCalla (2004) proposes that information should be gathered during the actual use of a Learning Resource and not only during explicit labelling phases; i.e., phases where metadata are created manually. In his so called ecological approach information about learners is captured during the use of documents and stored together with the documents. Thus, the documents become agents that can, linked with other agents, adapt to the learners' needs. In our terms this approach captures user centric context information. However, this approach only covers the actual usage of a document and neglects the other phases of a document's lifecycle.

Attention Metadata or *Contextualised Attention Metadata (CAM)* as described in CAMs (2008) and Wolpers et al. (2007) contains information about the attention a user pays to different Learning Resources via different applications. CAM is captured in

several applications and for various purposes. In existing approaches, CAM is, for instance, gathered and utilised to gain information about a user's experience (Najjar et al., 2006) or used for the ranking and retrieval of Learning Resources (Ochoa and Duval, 2006). In Wolpers et al. (2006) Attention Metadata is even used to combine learning management systems with organisational knowledge stores. In Chirita et al. (2005) contextual metadata are used to support desktop search engines like Beagle++ (Chirita et al., 2006). For capture and management of Attention Metadata, an approach similar to ours involving plug-ins and a central instance is used. Although very closely related there are two main differences to our approach: on the one hand, CAM is, as its name implies, meant for the capture of context information. So far, relation information is mostly neglected. On the other hand, the scheme is designed for information to be collected in a user centric way and not for the storage and processing of document centric information (see Lehmann et al., 2008a).

ALOCOM (Verbert et al., 2008) is a system that supports re-use of Learning Resources by means of a modular content model (Verbert et al., 2006). One part of the ALOCOM system is a PowerPoint add-in that helps users to re-use slides by enabling an online search in several repositories. In addition to that, it captures user behaviour (attention) in PowerPoint and provides it to the previously described Contextualised Attention Metadata framework. A Word add-in provides similar functionality but with the focus on Wikipedia as content source. When new documents are added to the system, ALOCOM decomposes them and captures structural relations for the newly added objects. Re-use is detected by comparing the new components with existing components in the repository by means of post-processing algorithms. Thus, it is possible to track re-use on component level. These relations are then used to, e.g., rank search results within ALOCOM. Klerkx et al. (2006) have examined and visualised re-use relations of PowerPoint components in ALOCOM. However, ALOCOM is not able to track relations other than on the component level and due to the fact that re-use is detected in the aftermath information gets lost. This includes, e.g., the time when a re-use operation occurred, or which document is the source and which document is the target of the re-use process.

In the APOSDLE project (Mayer, 2005) context information gathered from a user's desktop is used to determine the actual task a user is performing. This is done by means of machine learning approaches. The information about the current task is then used to recommend resources, documents or relevant experts that could help the user in her current task (Lokaiczyk et al., 2007b). Input used to determine the current task includes keyboard and mouse interaction, Clipboard changes, opened and closed documents, running applications or the user name (Lokaiczyk et al., 2007a). All the information from different sources is gathered and forms a 'context stream' which serves as input to the machine learning algorithms. In a first step the system is trained: Desktop context is monitored while users tell the system manually when a task switch occurs. Later on, the system automatically detects task switches. This approach collects context information in a user-centric way solely for the purpose of task detection. Nevertheless the technical means used to gather the information are similar to the ones used in LIS.KOM.

Mueller (2006) proposes a consistent management of change for structured as well as unstructured documents. The approach relies on a dedicated XML model and takes intra- document as well as inter-document relations into account. However, the approach does not aim to collect information but offers an explicit system for an improved versioning of documents.

In *TeNDaX* (Hodel et al., 2005), a system for collaborative creation and editing of text based documents, user actions are stored as transactions in a database. Thus, it is possible to track copy and paste relations between documents and draw relations accordingly. Nevertheless, to make this possible the TeNDAX editor has to be used.

4 Challenges of lifecycle information capturing

On our way towards integrative and transparent capturing of lifecycle information in office applications we had to cope with several challenges. We will discuss them here in general and use examples from Word and PowerPoint to further illustrate them whenever necessary. In Section 5 we will look at some of the challenges in greater detail and describe how we resolved them in the respective applications.

Get all relevant events that provide for the emergence of relations: It is a complex task to do information capturing within an office application without access to its source code. You have to rely upon its API to provide the means to properly monitor a user's interaction with the application. However, in Word as in PowerPoint the API only provides a small set of events that can be monitored directly, which were by no means sufficient for our purposes. In PowerPoint, e.g., the creation of a new slide triggers an event, whereas the deletion of a slide does not. In Word an event is raised whenever the save dialog is opened, but none when a document is actually saved. However, it is not sufficient to only monitor a user's actions within the respective application. Suppose a user copies a paragraph out of a PDF document into a Word document. This should trigger the creation of a relation between the PDF and Word document. Since this information is only available at the time when the user copies the paragraph from the PDF document to the system's Clipboard we have to capture it at the time the copy event occurs; i.e., we also have to monitor actions outside the respective applications.

Get all information needed to properly construct a relation: It has to be guaranteed that enough information is available to construct a valid relation; i.e., we have to know the source and the target document and, depending on the event type, additional information about the respective elements within the documents. Furthermore, we need context information and the relation type to construct a valid relation. When, for instance, an external object is re-used (e.g., a paragraph is copied from a PDF document to a Word document) the API usually does not provide all the information needed to construct a valid relation. In a few cases the system's Clipboard stores additional information, but mostly other techniques to retrieve all necessary data have to be found.

Tracking and validation of existing relations: An established relation is not valid indefinitely. When a slide is re-used in a new presentation the re-use relation has to be removed if that slide is deleted in either the original or the re-using presentation. Hence, mechanisms have to be developed that detect when established relations are no longer valid. However, the decision about whether or not a relation is still valid is not always as easy to decide as in this example. If a user does not delete an entire slide but only a couple of shapes or a user copies a paragraph from one document to another and starts rephrasing it, then the decision whether or not an established relation is still valid cannot be decided in general; i.e., a case to case decision has to be made using

fuzzy logic approaches. It has to be decided where to draw the border between "the relation is still valid" and "the relation is invalid" since no universal rule exists.

Handle changes done by users without the respective add-ins: We usually cannot assume a closed scenario where every user has the required add-ins installed. Thus, we have to handle changes to the content of a document, which we have not been able to monitor. A minimum requirement is that the collected information is kept valid.

Handle events and changes occurring while the office application is not running: We have to cope with changes that occur while the respective office applications are not running, since an add-in naturally only works while its application is running. This includes changes in the file system, like renaming, deletion or movement of involved files. It becomes more complicated when users, e.g., share documents via e-mail or file-sharing services.

Handling document variants: Whenever a document is copied, sent via e-mail or 'saved as' a new document instance, a document variant is created. In many cases these document variants are closely related to the original document. However, we should able to tell whether this is the case or not.

5 Implementation

In this section we describe our LIS.KOM Framework and how it handles lifecycle information capturing. We will then show how we resolved the challenges described in Section 4 by means of add-ins for PowerPoint and Word. We start with a description of approaches that we used in Word and in PowerPoint, and that we deem generally applicable. We will then describe the specifics and differences for both implementations.

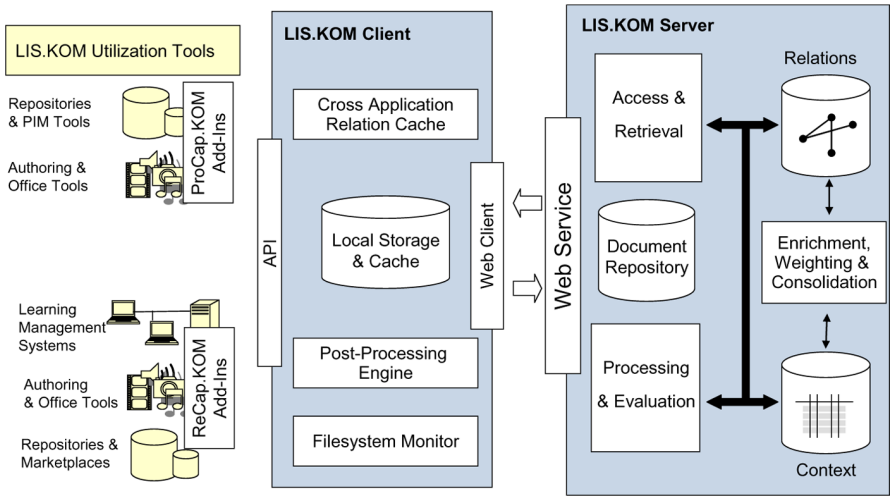
5.1 The LIS.KOM framework

The LIS.KOM Framework was designed to enable the capture, management, and utilisation of lifecycle information. The easiest way to describe the LIS.KOM Framework is to follow the flow of information through the system (see Figure 3). In order to obtain lifecycle information it first has to be captured. Due to lifecycle information being process-oriented it has to be captured during the respective processes, usually within the respective applications. This is done by means of so called ReCap.KOM add-ins which are the main focus of this paper. For each supported application there is one add-in to collect the information. The captured information is then sent to the LIS.KOM Client where it is cached, processed and validated (see Sections 5.2–5.4 for details).

Via a web service interface the LIS.KOM Client connects to the central LIS.KOM Server to synchronise the lifecycle information collected on the client side. Since this is done by all the clients participating in the system, the LIS.KOM Server, thereby, aggregates all captured lifecycle information. On the server side, the information is aggregated, validated and prepared for utilisation. This can include enrichment (e.g., enrichment of context information with relation information), consolidation and weighting. Consolidation and weighting becomes important, e.g., when there are several relations between the same two documents; e.g., because several slides have been copied between two presentations. In this case, these relations can be subsumed under one

relation but have to be weighted accordingly. Afterwards the processed information can be obtained via the web service interface, in order to be utilised. Additionally, the server provides the possibility to store the documents themselves instead of the lifecycle information only. Thus, users can be provided with the respective documents when utilising the lifecycle information.

Figure 3 LIS.KOM framework (see online version for colours)



5.2 Common capturing strategies

Most of the challenges identified in Section 4 can be solved using the same methods in different applications. In the following we describe the techniques we use to achieve this. The APIs of office applications (like e.g., Microsoft Office) are usually not meant to allow for easy monitoring of the actions that a user takes while creating new documents, but to provide a means for automated document creation. Hence, the amount of events raised upon user actions is quite small. In order to receive enough events to correctly capture re-use relations, existing events have to be combined with low-level mouse and keyboard hooks as well as with a monitoring service that raises events on Clipboard activity. In Word as in PowerPoint we get events when specific buttons or menu items are activated (e.g., 'Insert', 'Copy', 'Delete', 'Undo', etc.). However, while PowerPoint, for example, offers reliable events if a presentation was saved, Word does not. To work around this, we constantly check the state of opened documents and raise our own 'Save' and 'Save As' events. The keyboard hook triggers method calls when specific keys or key combinations ('CTRL+C', 'CTRL+V' etc.) are pressed while the Clipboard monitor notifies us about changes on the Clipboard. With this combination of native and custom events it is possible to monitor most of the actions relevant to the capture of re-use relations. When monitoring low-level events we have to be able to tell where they occurred. We achieve this by using the operating system's native API to tell which application and document is currently active.

The information needed to construct a relation includes information about the source and target document (IDs, Path/URL, Title, Owner etc.), the relation type, a timestamp and for most relations the specific elements within the source and target document that are connected through the relation. Basically, there are three scenarios that require different measures to obtain this information:

1 Source and target document both come from one application that we monitor (e.g., content is copied between two PowerPoint presentations).

2 Source and target document come from different applications both of which we monitor (e.g., content is copied from Word to PowerPoint).

3 Content is copied from an external source to a document in an application that we monitor (e.g., from a website to a Word document).

In the first case we get the information directly from the add-in. If an object or text is copy-pasted between applications which both have a ReCap.KOM add-in, the relevant information can be retrieved by the add-ins but it has to be transferred between the two application contexts. This is done using the Cross Application Relation Cache (CARC). Every time an object is copied, the source information of this object is stored along with its unique hash in the CARC. When something is pasted in a monitored application the source information for the pasted object is obtained from the CARC by its hash value. Thus, a fully-fledged relation can be constructed and stored in the local lifecycle information storage of the LIS.KOM Client.

If content is copied from an external object, we have two choices: In some cases, for example, if text is copied from a browser to the Clipboard, the Clipboard contains all relevant information (in this case, the URI of the website the text was taken from as well as the text itself). In most cases however, the Clipboard will only contain the copied content. Here we use simple screen scraping approaches to infer the element's origin. In many cases, the window title contains the name of the currently opened file. If so Window's 'Recent Folder' provides an almost perfect lookup table to the file's absolute path, since most applications store a link to each opened file in this special folder. If this is not the case (e.g., a remote PDF file is opened inside a browser), hints to the file's origin can usually be found somewhere on the screen and be captured with custom screen scraping approaches (in the browser example the relevant information can be obtained from scraping the browser's location bar).

For capturing and validation of document variants we make use of high-similarity-near-distance fingerprints. In contrast to typical (cryptographic) hashes, these fingerprints change only marginally when the underlying documents change. We have customised and extended existing approaches (like e.g., Charikar, 2002; Broder et al., 2000) which yield good results for textual documents, to fit our needs. Each document is assigned a fingerprint, which subsumes its characteristics. Thus, we are able to tell whether documents are variants very efficiently and without needing access to their content.

As stated in Section 4, we need a mechanism to validate captured relations. To achieve this we have customised and extended fuzzy string matching approaches known from the plagiarism search area (e.g., Lyon et al., 2001; Wise, 1993).

Lastly, changes that occur while Word or PowerPoint are not running need to be captured, too. Specifically, when document names or paths are changed LIS.KOM needs to be notified to update the information accordingly. We do this by means of a

component called Filesystem Monitor which is constantly running in the background and triggers events when files of specific types are renamed, copied, deleted or moved.

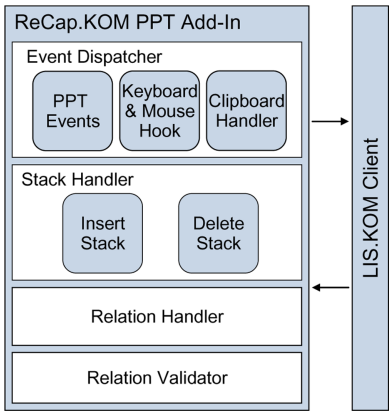
Hence, almost all of the challenges posted in Section 4 can be handled by means of application independent approaches. The only issue handled fundamentally different in Word and PowerPoint is keeping track of the consistency of the captured information. This and how the add-ins for PowerPoint and Word are structured in detail we describe in the following subsections.

5.3 PowerPoint add-in

To keep the actual information state in the local storage synchronous with the actual document state we make use of a tracking approach in PowerPoint. The PowerPoint data model is quite modular and almost everything can be accessed via an ID. Additionally, PowerPoint provides enough information and events to always keep track of a user's actions. In order to keep the information captured consistent, delete, undo and redo actions of users have to be monitored. Through deletion of content a captured relation may become invalid. Undo and redo actions have to be taken into consideration since they can render a just established relation invalid or make a deleted relation valid again. Since the given API does not provide direct access to the undo and redo stacks we implemented our own solution. The Stack Handler handles all actions relevant for the actual lifecycle information state. Since errors in the stack handler propagate, we use validation techniques as fallback solution.

The components of the ReCap.KOM add-in for PowerPoint are depicted in Figure 4. In PowerPoint most events come from the application itself. However the Clipboard is a reliable source for 'copy' events. The keyboard is only monitored to capture shortcuts or the 'delete' key in the PowerPoint case.

Figure 4 PowerPoint add-in (see online version for colours)

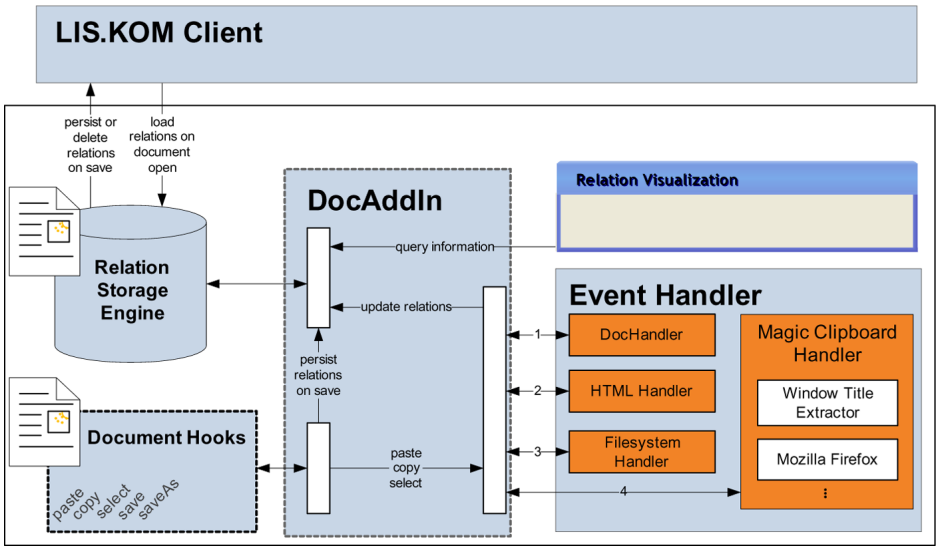


5.4 Word add-in

Figure 5 depicts the main elements of the ReCap.KOM add-in for Microsoft Word. Next to low-level keyboard hooks, we had to implement custom 'save' and 'save as' events, since Word does not reliably raise events when documents are saved. We achieve this by constantly testing the status of all opened documents. All captured events are

passed on to an event handler chain, consisting of several modules that try to infer established relations from the raised events. Basically, these implement the different approaches described earlier. Captured relations are then cached in memory in the relation storage engine.

Figure 5 Components of word add-in (see online version for colours)



Due to the lack of events raised and the lack of a modular data model in Word it is impossible to keep the state of captured information always synchronous to the actual document state. Thus, we cannot use the tracking approach we use for PowerPoint. To work around this, we use a validation approach; i.e., we validate all relations whenever a document is saved. Relations that are still valid are persisted in the local lifecycle information storage of the LIS.KOM Client. An evaluation will show how accurately this approach works. Since the relation information is validated every time a document is saved we do not need to validate information at any other stage. To enable validation of captured relations, in case of text reuse, we not only capture the information needed to construct the relation (like e.g., the target's and source's document IDs) but also the copied text itself. Thus, we are able to not only identify if a relation still holds but also where it points to within the document.

On validation, the captured text is compared to the corresponding document text to judge the validity of the respective relation using the aforementioned, custom, high-performance string matching algorithm.

6 Evaluation

We have conducted a first evaluation to show the feasibility of our approach and published the results in Lehmann et al. (2008b). Thus, we will only summarise the results here. The goal of this evaluation was to prove the validity of lifecycle information captured with our framework. We deployed a ReCap.KOM capture module for

PowerPoint on the computers of four test persons. In this evaluation the following types of relations were captured:

- Variant relations (isVariantOf/hasVariant)
- Re-use of media objects from the file system (isPartOf/hasPart)
- PowerPoint internal re-use of content (providesElementTo/ContainsElementOf).

The evaluation ran for about six weeks in a realistic usage scenario, i.e., the test persons used PowerPoint as they would have without being test persons. In this evaluation lifecycle information capturing was done locally without involvement of the central LIS.KOM Server. Table 1 shows the results of the evaluation with the types of relations captured, their respective quantity, the distribution among the test persons and their validity. The validity of a relation was determined by manually traversing the captured relations and judging whether the collected information was valid or not. A relation was rated as inconclusive when its validity could not be judged because the document the relation pointed to had been deleted. This could happen since the test persons were not forced to keep all documents. Altogether there were 58 relations of PowerPoint internal re-use for 29 different documents. This shows that there is a high amount of reuse happening when PowerPoint presentations are created.

Table 1 Captured relations and their validity

Relation type	*Total*	*Distribution* (*test person 1–4*)				*Valid*	*Invalid or inconclusive*
Provision (total)	58	3	5	0	50	43	15
Prov. – slides	38	3	0	0	35	29	9
Prov. – text	12	0	5	0	7	8	4
Prov. – shapes	8	0	0	0	8	6	2
Variant	16	2	4	1	9	16	0
Media objects	23	18	0	5	0	23	0

Source: Lehmann et al. (2008b)

In this evaluation we have captured relations with an overall validity of about 85%. The main reasons for relations being invalid were:

1 The evaluation scenario was not closed. People without ReCap.KOM add-ins edited some of the presentations and the validation algorithm was not part of the evaluation.

2 We have discovered a minor event handling issue that is responsible for some of the invalid relations (we have fixed it meanwhile).

3 We counted a relation as invalid even if it was the result of a re-use process, but there was no similarity in content. This for instance happens if slides are re-used for structural reasons (to have a ‘template’).

The two main findings of this evaluation were that

1. re-use is a common practise when PowerPoint presentations are created and that
2. the capture of relations is feasible.

Currently we are planning and preparing an evaluation of the whole LIS.KOM system. We plan to have 10–20 client systems taking part in the evaluation, each with PowerPoint and Word add-ins as well as LIS.KOM Clients installed. The captured information will be collected, enriched and consolidated on the central LIS.KOM Server. The server enables us to capture re-use of content of different authors as well as personal re-use. We plan to run the capturing for about 8 weeks. Again, we do not want to impose any restrictions on the authors taking part in the evaluation. Therefore, the scenario will not be closed and the authors may handle their documents as usual (although they are asked to delete as few documents as possible). We will keep all the documents involved in any relation that was captured, in a document pool for different evaluation steps.

We want to evaluate the following aspects:

- *Validity of captured information*: The documents of a relation will be examined to judge whether the collected relation is valid or not. This has to be done manually.
- *Comparison with post-processing approaches*: An alternative approach to ours would be the acquisition of information automatically in the aftermath. To compare our approach with such approaches, state of the art post-processing algorithms (like e.g., k-gram) will be implemented and applied to the document pool. Then we will compare the information captured with our approach with the information obtained via post-processing algorithms with regard to quality (i.e., validity) and quantity.
- *Comparison with ground truth*: Another possibility is the construction of a ground truth out of the document pool to rate our approach as well as the post-processing algorithms.

In addition to these aspects, the evaluation will provide further insights into, e.g., the re-use behaviour of authors of office documents. We plan to visualise the captured relations between documents along with the corresponding authors to show how documents (and authors) are connected.

7 Conclusions and future work

The management and retrieval of office documents suffers from the lack of information and metadata about these documents. In this paper we have proposed the capture of information in office applications as a means to automatically generate information that is utilisable for search, retrieval or authoring of office documents. We identified the main challenges and described how they have been overcome in our LIS.KOM Framework, which can be extended by add-ins for applications where information should be collected in. We have exemplarily implemented two add-ins for applications with different preconditions – Word and PowerPoint. PowerPoint is an example for an application where the events provided by the API enable a decent monitoring of user actions. Word is an example of an application where the number of events provided by the application is very sparse and the monitoring has to be done almost solely via events provided by the operating system. We have shown that the capture of lifecycle information is feasible in both cases. Thus, our approach is transferable to arbitrary applications as long as very basic functionality and events are provided.

The first evaluation we have conducted has shown that re-use is common practice and that relation information can be captured with high validity. The upcoming evaluation of the whole LIS.KOM Framework will show the feasibility of lifecycle information capture in a community scenario, how well information capture performs compared with post-processing approaches, and provide insights into the re-use behaviour of authors of office documents. The most important step beyond this evaluation will be the utilisation of the gathered information. Besides the application specific add-ins that provide information about related documents and can notify authors of updates, we plan to implement a standalone application for search, browsing and retrieval of documents. User evaluations have to be conducted to judge if the collected information and the way it is utilised in these applications is helpful.

References

Mayer, H., Haas, W., Thallinger, G., Lindstaedt, S. and Tochtermann, K. (2005) 'APOSDLE – advanced process-oriented self-directed learning environment', *EWIMT 2005: 2nd European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies*.

Broder, A.Z., Charikar, M., Frieze, A.M. and Mitzenmacher, M. (2000) 'Min-wise independent permutations', *Journal of Computer and System Sciences*, Vol. 60, No. 3, pp.630–659.

CAMs (2008) *Contextualized Attention Metadata Conceptual Base Scheme*, Online.

Charikar, M.S. (2002) 'Similarity estimation techniques from rounding algorithms', *STOC'02: Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing, ACM*, New York, NY, USA, pp.380–388, ISBN 1-58113-495-9.

Chirita, P-A., Costache, S., Nejdl, W. and Paiu, R. (2006) 'Beagle++: semantically enhanced searching and ranking on the desktop', *The Semantic Web: Research and Applications*, Springer, Berlin/Heidelberg, pp.348–362.

Chirita, P-A., Gavriloaie, R., Ghita, S., Nejdl, W. and Paiu, R. (2005) 'Activity based metadata for semantic desktop search', *Proceedings of the 2nd European Semantic Web Conference*, Heraklion, Crete, Greece, pp.439–454.

Duval, E. and Hodgins, W. (2003) 'A LOM research agenda', *Proceedings of the Twelfth International Conference on World Wide Web*, Budapest, Hungary.

Hodel, T., Hacmac, R. and Dittrich, K.R. (2005) 'Using text editing creation time meta data for document management', *Proceedings of Conference on Advanced Information Systems Engineering*, Porto, Portugal, pp.105–118.

Klerkx, J., Verbert, K. and Duval, E. (2006) 'Visualizing reuse: more than meets the eye', *Proceedings of the 6th International Conference on Knowledge Management (I-KNOW)*, Graz, Austria, pp.489–497.

Lehmann, L., Hildebrandt, T., Rensing, C. and Steinmetz, R. (2008a) 'Capture, management and utilization of lifecycle information for learning resources', *IEEE Transactions on Learning Technologies*, Vol. 1, No. 1, pp.75–87.

Lehmann, L., Rensing, C. and Steinmetz, R. (2008b) 'Capture of lifecycle information to support personal information management', in Specht, M. and Dillenbourg, P. (Eds.): *Times of Convergence: Technologies Across Learning Contexts, Third European Conference on Technology Enhanced Learning, EC-TEL 2008*, Springer, Crete, Greece, pp.216–221.

Libbrecht, P. (2008) 'A model of re-use of e-learning content', in Dillenbourg, P. and Specht, M. (Eds.): *EC-TEL*, Springer, Maastricht, Netherlands, pp.222–233.

Lokaiczyk, R., Faatz, A., Beckhaus, A. and Görtz, M. (2007a) 'Enhancing just-in-time e-learning through machine learning on desktop context sensors', in Kokinov, B.N., Richardson, D.C., Roth-Berghofer, T. and Vieu, L. (Eds.): CONTEXT, Springer, Roskilde University, Denmark, pp.330–341.

Lokaiczyk, R., Godehardt, E., Faatz, A., Görtz, M., Kienle, A., Wessner, M. and Ulbrich, A. (2007b) 'Exploiting context information for identification of relevant experts in collaborative workplace-embedded e-learning environments', in Duval, E., Klamma, R. and Wolpers, M. (Eds.): *EC-TEL*, Springer, Crete, Greece, pp.217–231.

Lyon, C., Malcolm, J. and Dickerson, B. (2001) 'Detecting short passages of similar text in large document collections', in Lee, L. and Harman, D. (Eds.): *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, Pittsburg, PA, USA, pp.118–125.

McCalla, G. (2004) 'The ecological approach to the design of e-learning environments: purpose-based capture and use of information about learners', *Journal of Interactive Media in Education*, Vol. 7, pp.171–193.

Meire, M., Ochoa, X. and Duval, E. (2007) 'SAmgI: automatic metadata generation 2.0', *Proceedings of World Conference on Educational Multimedia*, Hypermedia, Vancouver, Canada, pp.1195–1204.

Mueller, N. (2006) 'An ontology-driven management of change', *Wissens- und Erfahrungsmanagement, LWA (Lernen, Wissensentdeckung, Adaptivitaet) Conference Proceedings*, Hildesheim, Germany, pp.186–193.

Najjar, J., Wolpers, M. and Duval, E. (2006) 'Towards effective usage-based learning applications: track and learn from users experience(s)', *Proceedings of the IEEE ICALT 2006*, July, Kerkrade, Netherlands, pp.1022–1024.

Nejdl, W. and Paiu, R. (2005) 'I know I stored it somewhere – contextual information and ranking on our desktop', *Proceedings of 8th International Workshop of the EU DELOS Network of Excellence on Future Digital Library Management Systems*, Schloß Dagstuhl, Germany, pp.118–124.

Ochoa, X. and Duval, E. (2006) 'Use of contextualized attention metadata for ranking and recommending learning objects', *Proceedings of the CAMA 2006*, Arlington, VA, USA, pp.9–16.

Rensing, C., Bergsträßer, S., Hildebrandt, T., Meyer, M., Zimmermann, B., Faatz, A., Lehmann, L. and Steinmetz, R. (2005) *Re-Use and Re-Authoring of Learning Resources – Definitions and Examples*, Technical Report, KOM-TR-2005-02, TU Darmstadt – Multimedia Communications Lab, Darmstadt.

Verbert, K., Jovanovic, J., Duval, E., Gasevic, D. and Meire, M. (2006) 'Ontology-based learning content repurposing: the ALOCoM framework', *International Journal on E-Learning*, Vol. 5, No. 1, pp.67–74.

Verbert, K., Ochoa, X. and Duval, E. (2008) 'The ALOCOM framework: towards scalable content reuse', *Journal of Digital Information*, Vol. 9.

Wise, M.J. (1993) *String Similarity via Greedy String Tiling and Running Karp-Rabin Matching*, ftp://ftp.cs.su.oz.au/michaelw/doc/RKR GST.ps

Wolpers, M., Martin, G., Najjar, J. and Duval, E. (2006) 'Attention metadata in knowledge and learning management', *Proceedings of I-Know 2006*, Graz, Austria, pp.403–410.

Wolpers, M., Najjar, J., Verbert, K. and Duval, E. (2007) 'Tracking actual usage: the attention metadata approach', *International Journal Educational Technology and Society*, Vol. 11, pp.106–121.