The ALOCOM Framework: Towards Scalable Content Reuse

Katrien Verbert¹, Xavier Ochoa², and Erik Duval¹

¹Dept. Computerwetenschappen, Katholieke Universiteit Leuven
Celestijnenlaan 200 A, B-3001 Leuven, Belgium
{katrien.verbert, erik.duval}@cs.kuleuven.be
²Escuela Superior Politécnica del Litoral, Campus "Gustavo Galindo"
Guayaquil, Ecuador
xavier@cti.espol.edu.ec

Abstract. This paper presents the ALOCOM framework for flexible content reuse. Unlike the usual practice where document components, such as images, definitions, text fragments, tables or diagrams, are assembled manually through copy-and-paste, the framework enables on-the-fly access and reuse. Retrieval of relevant components is enabled by automatic decomposition of legacy documents and storage of individual components, enriched with metadata. Furthermore, the automatic assembly of these components in mainstream authoring tools is supported. The paper describes the framework and its current support for re-assembling PowerPoint, Wikipedia and SCORM components in authoring tools. In addition, an evaluation is presented that assessed the effectiveness and efficiency of such content reuse for presentations.

1. Introduction

In document engineering, the transformation of documents into forms that ease their reuse is an important research topic (Lecerf & Chidlovskii 2006). The practice of reuse has significant advantages for document creation, maintenance and use. It saves work, improves document quality and increases consistency by eliminating re-doing what has been done before and allowing the user to make a change in one place instead of many (Barta & Gil 1996).

In non-fiction writing, it is a common practice to construct documents from pieces of existing material (Barta & Gil 1996). In the software industry, manuals for a new software release are built from previous versions; in legal writing, adaptation and reuse of previous documents is a universal practice, as lawyers often assemble documents, such as contracts, from existing text templates, filling in a few blanks (Branting & Lester 1996). Also in the pedagogical domain, there is an increasing demand for reusing Learning Objects (Downes 2001, Robson 2004, Duval & Hodgins 2003).

A lot of research is dedicated to re-assemble reusable content components, often referred to as single sourcing (Ament 2003). Single sourcing implies that there is a single source for content: content is written once, stored in a single source location, and reused many times (Rockley 2002). Recent research focuses on the dynamic assembly of content components into personalized documents (Vercoustre & McLean 2005).
However, little work has been done to support automatic decomposition of legacy documents. Instead, guidelines (Schluep 2005) are often provided to decompose content manually or to create new components that are suitable for reuse. The approaches focus on optimizing reuse of content that has been specifically designed for this purpose, however, are unable to scale. If we can decompose documents automatically into reusable components, reuse of numerous document components available on the World Wide Web can be automated.

There are a number of issues that need to be dealt with to realize the approach:

1. First of all, there is the question of how far it is useful to decompose documents into components. As pointed out by Rockley (2002), sentence fragments or individual words may not be appropriate for reuse, as their added value for reuse is questionable. Complete sections or chapters on the other hand may be too coarse-grained for use in a different context. The issue of finding an appropriate granularity level for decomposition is further described in Section 2.1.

2. Secondly, as decomposition requires explicit structure, the transformation of semi-structured or unstructured documents into an explicitly structured format needs to be supported (Lenski & Wette-Roch 2001).

3. Thirdly, because authors will rely on mainstream authoring tools, integration of assembling content components into the workflow of authors needs to be examined (Kienreich et al. 2005).

In earlier work, we have presented an abstract content model (ALOCOM) for documents and their components (Jovanović et al. 2005). The model defines content component at different levels of granularity and relationships between components. As such, the model enables structuring of composite documents and is a solid basis for the proposed automated reuse approach.

This paper presents the ALOCOM framework for both decomposition and re-assembly of document components. The framework supports extracting content components from existing documents. In addition, reuse is detected to avoid duplicates and metadata are automatically added to components. Prototypes validate the approach for presentations, Wikipedia pages and SCORM content packages (SCORM 2004). Plug-ins for Microsoft PowerPoint, Microsoft Word and the Reload Editor (Milligan et al. 2005), a packaging tool designed for composition of SCORM content packages, enable authors to search and reuse components from within the authoring tools.

We have carried out a user evaluation that assessed the usability of the plug-in for Microsoft PowerPoint. The goals of the evaluation were threefold: (i) to assess the efficiency and effectiveness of the approach for reusing presentations; (ii) to assess the subjective acceptance of the ALOCOM interface; and (iii) to determine to which level of granularity decomposition is relevant. A follow-up evaluation was necessary to confirm the results and assessed the quality of the created presentations.

The paper is organized as follows: The next section presents an overview of the ALOCOM framework. Section 3 describes the Microsoft PowerPoint, Microsoft Word and Reload plug-ins that support the aggregation process. User evaluation results are presented in Section 4. The quality evaluation is presented in Section 5, followed by a discussion in Section 6. Related work is described in Section 7. Conclusions and remarks on future work conclude this paper.
2. The ALOCOM Framework

Reuse typically requires a library of assets that have been enriched with metadata (Rada 2001). The ALOCOM framework supports such reuse by decomposing pre-existing documents into smaller, reusable, components and storing the components individually. Furthermore, on-the-fly access to these components is provided. The server relies on the ARIADNE Knowledge Pool System (Duval et al. 2001) for storage of components and their metadata. The framework is depicted in Figure 1 and consists of the following components:

1. Client side applications (Figure 1 - (1)) within authoring tools that enable content uploading to and component retrieval from the repository. Plug-ins have been developed that provide these functionalities for Microsoft PowerPoint and Reload. A plug-in for Microsoft Word enables automatic reuse of Wikipedia components in text documents (see Section 3).

![Figure 1. The ALOCOM Framework](image-url)
2. The Disaggregation module (Figure 1 - (2)) supports the actual decomposition. Presentations are decomposed into slides, and slides are further decomposed into images, tables, diagrams, audio and video sequences, and text fragments. Text documents are decomposed into sections and subsections, and each section is further decomposed into paragraphs, images, tables, diagrams, etc. The current implementation of this module supports the approach for PowerPoint presentations and Wikipedia pages. Components are extracted, preview thumbnails are generated and results are stored through the AdvancedContentInserter (see Section 2.1).

3. The AdvancedContentInserter (Figure 1 - (3)) provides support for storing not only complete documents, but also components that are contained in the document, for instance components stored in a SCORM content package or components that were extracted by the Disaggregation module. The AdvancedContentInserter supports reuse detection for different component types, adds metadata to each component, and stores the components and preview thumbnails in the repository (see Section 2.2).

4. The Ranking module (Figure 1 - (4)) assigns ordering values to components based on their reuse and enables ranking of components in result lists when a user searches for relevant objects, placing components with a high relevancy at the top of the list (see Section 2.3).

5. The Query Service (Figure 1 - (5)) enables retrieval of components. Both descriptive keywords and a component type, such as definition, example, slide, image, diagram or table, can be specified when searching for components. Also advanced queries are supported that enable searching by author, title, main concepts, duration, etc. (see Section 2.2.3).

The rest of this section details the server components. The client applications are described in Section 3.

2.1 Disaggregation Module

The disaggregation module automates decomposition of composite documents into components. Granularity is an important factor in this process. The size of a component can vary between a chapter and a single line. The more fine-grained the structure is, the more flexible possibilities for document reuse are obtained. However, more fine-grained also results in a larger set of components and is more complex to manage (Dahn 2001, Rockley 2002). As pointed out by Rockley (2002), sentence fragments or individual words may not be appropriate for reuse. However, single paragraphs may constitute definitions, examples or exercises that are reusable. That is why we decompose to the level of paragraphs. For the approach to remain scalable, modules for detection of reuse, generation of accurate metadata and ranking are incorporated in the framework (see Section 2.2.1, 2.2.2 and 2.3 respectively).

Figure 2 illustrates the decomposition process. Presentations are decomposed into slides, and slides are further decomposed into images, tables, diagrams, animations, audio and video sequences and text fragments. Text documents are decomposed into sections and subsections, and each section is further decomposed into paragraphs, images, tables, diagrams, etc. The type of information contained in components, such as definition or example, and other relevant information for component retrieval, is determined by the automatic metadata generation module (see Section 2.2.2).

The current implementation of this module automates decomposition of Microsoft PowerPoint presentations and Wikipedia pages. The latter are decomposed on-the-fly at the client side (see Section 3.2). Decomposition of presentations is performed on the server. The module is implemented as a .Net web service and uses the PowerPoint API (Khor & Leonard 2005) to retrieve content and structure from a presentation.
The decomposition method iterates over the slides and slide shapes of a PowerPoint presentation object. Each slide is stored in the PowerPoint format to enable lossless reuse. Images are extracted and stored in their original format and text fragments and tables are stored in an XML format containing their content and structure. For slides, an XML representation is generated to enable their reuse in other applications and for detecting reuse between slides (see Section 2.2.1). Finally, preview thumbnails are generated for each component, using built-in export functions provided by the PowerPoint API. These thumbnails are used in the search interface of client applications (see Section 3.1).

In the next step, the generated components are sent to the AdvancedContentInserter for storage and indexation.

2.2 AdvancedContentInserter

The AdvancedContentInserter is part of a Java web service that relies on the ARIADNE Knowledge Pool System (Duval et al. 2001) for storage of document components. The module automates reuse detection for individual components and metadata are added by an extended version of the Automatic Metadata Generation (AMG) framework (Cardinaels et al. 2005).

Individual components can be sent to this service, provided with the identifier of the parent component, but also complete packages, such as SCORM (SCORM 2004), METS (Cundiff 2004), or MPEG-21 (Burnett et al. 2005) content packages. Each of them is based on the idea of a central XML manifest file and either references or contains the data files that make up the package. In the latter case, the AdvancedContentInserter processes the manifest file and components are stored individually.

2.2.1 Reuse Detection

Components that are reused in different documents would result in duplicate components in the repository. Reuse detection is used to avoid these duplicates.

The problem of finding near-duplicate documents has been a subject of research in the database and web-search communities for some years (Yang & Callan 2006). The applications range from
plagiarism detection in web publishing (Hoad & Zobel 2003, Shivakumar & Garcia-Molina 1995) to redundancy detection in large datasets (Foo et al. 2007, Yan et al. 2004). The common duplicate detection techniques are classified into two categories: Full text-based and Fingerprint-based. The approaches are briefly presented in the rest of this section, followed by a brief discussion of their application in ALOCOM.

2.2.1.1 Duplicate Detection Using Full Text

Full text-based approaches use methods originally developed for search engines. An example of such an approach is the vector-space model, which treats a document as a bag-of-words. Similarity is commonly determined by the cosine similarity measure (Shivakumar & Garcia-Molina 1995, Yang & Callan 2005). The approach represents each document as a vector in an n-dimensional space. Document similarity to another document is then defined as the distance between the two vectors:

\[
sim(R, Q) = \frac{\sum_{i=1}^{N} \alpha_i^2 \cdot F_i(R) \cdot F_i(Q)}{\sqrt{\sum_{i=1}^{N} \alpha_i^2 \cdot F_i^2(R) \cdot \sum_{i=1}^{N} \alpha_i^2 \cdot F_i^2(Q)}}
\]

where \( \alpha_i \) is the weight associated with the occurrence of the \( i^{th} \) word and \( F(D) \) (size n) is the frequency vector. \( F_i(D) \) is the number of occurrences of word \( w_i \) in text fragment \( D \).

To illustrate the similarity computation, consider a registered text fragment \( R = "a \ b \ c" \) and new text fragments \( S_1 = "a \ b \ c" \) and \( S_2 = "c \ d \ e" \). Using the cosine similarity measure for the example and assuming uniform weights for words (\( \alpha = 1 \)),

\[
sim(R, S_1) = \frac{1 \cdot 1 \cdot 1 \cdot 1}{\sqrt{3 \cdot 3}} = 1 \text{ and } sim(R, S_2) = \frac{0 \cdot 1 \cdot 0 \cdot 1 \cdot 1}{\sqrt{3 \cdot 3}} = 0.3.
\]

Identical text fragments have a similarity value 1, while text fragments that do not have much overlap have a low value (e.g. 0.3 in the example).

The approach has been extended by researchers to incorporate term frequencies (HE & Ounis, 2003), document length (Singhal et al. 1996), and term frequency/inverse document frequency weighting (Salton & Buckley 1988). Such extensions improve the effectiveness of similarity comparisons.

2.2.1.2 Duplicate Detection Using Fingerprints

A second commonly used approach is based on fingerprints. A fingerprint of a document is a set of integers, each of which is the hash value for a substring or image extracted from the document (Yang & Callan 2006). Similarity between two documents is then measured by counting the number of common integers. Algorithms are different in their choices of hash functions, substring size, substring number, and substring selection strategy:

- Several commonly used hash functions are MD5 and SHA-1 (Manku et al. 2007). These functions have three desirable properties for duplicate detection: they can be calculated on arbitrary document lengths, are easy to compute, and have very low probabilities of collisions.
- Substring size is defined as the length of each substring extracted from a document. Some approaches use single words as the unit for fingerprinting (Shivakumar & Garcia-Molina 1995), whereas others use full sentences (Brin et al. 1995).
- Substring number is the number of substrings extracted from a document to build a fingerprint. Some techniques use a fixed number of substrings for efficiency, while many others use a variable number of substrings for a more accurate representation of the document.
• Substring selection strategy is the way to select which substrings to hash. A position-based approach, for instance, selects substrings based on their offsets in a document.

2.2.1.3 Reuse detection in ALOCOM

We use a full text-based approach for detecting reuse between text fragments. A Lucene index (Gospodnetic & Hatcher 2005) is used for storage of text fragments. New text fragments are compared against fragments stored in the index. The cosine similarity measure is used for overlap computation. As the text fragments are fine-grained, the technique works well. A fingerprint-based approach can be employed when dealing with larger documents or for improving efficiency.

For images, a duplicate detection technique has been implemented that uses the MD5 hash value of the file. This technique enables to detect identical images. As adaptations to images are much less frequent than adaptations to text fragments, the current technique is working reasonably well. The approach can be improved by incorporating techniques for detecting near-duplicate images, that are often based on color histograms or local feature descriptors of images (Yan et al. 2004, Foo et al. 2007). Incorporating such an approach would enable to detect common transformations, such as changing contrast, saturation, scaling, cropping, and framing.

2.2.2 Metadata Generation

During decomposition, metadata are added to each component. As manual provision of metadata remains tedious and cannot scale, we need flexible and scalable ways to generate and extract metadata from content and context (Handschuh 2002, Greenberg 2004). In our research unit, a framework has been developed for such automatic metadata generation (Cardinaels et al. 2005). The idea behind the framework is to combine metadata, generated from different sources, in a single metadata instance. The first source is the document itself; the second is the context in which the document is used. Metadata derived from the document is obtained by content analysis, such as keyword extraction and language classification. The contexts typically are content management systems or author institution information.

Additional information gained by the decomposition process is used in the annotation process. For instance, Microsoft PowerPoint provides “place holders” to type the title of a slide. This title is added to the metadata instance as the title of the component.

Furthermore, the metadata describing a component can also be deduced from the metadata for its parents. For instance, each slide in a presentation inherits the author, language, etc. from the presentation to which it belongs. For this purpose, an extension of the framework has been developed that combines metadata by an inheritance mechanism.

Finally, dependency relations between document components are described as relationship metadata. Through additional attributes, we can distinguish different relations between parent and child components (“isPartOf”, “hasPart”) and between components (“ordering”).

2.2.3 The Query and Insert Service

The ARIADNE query service (Ternier et al. 2003) is used for retrieval of content components and the insert service for inserting components in an ARIADNE Knowledge Pool (Duval et al. 2001). The insert service supports inserting, updating and deleting components and their metadata.

The ALOCOM repository is currently filled with 62841 components that were extracted from 814 documents. These components include 18149 slides, 7028 images, 226 tables, 30 diagrams and 35460 text fragments.
2.3 Ranking

Document decomposition results in a repository filled with numerous components. Hence, there is a need for a ranking mechanism, so that searches are only confronted with relevant components and the approach remains scalable. The ranking function assigns a value to a component based on three metrics:

- the number of times the component has been reused directly,
- the number of times the component has been reused as part of a bigger component,
- and the number of different authors that have reused the component.

While these metrics measure the historical probability that a component will be reused, a more useful approach is to calculate the probability that a component will be selected on a specific date. This probability is calculated providing a time frame, for example the previous month or year, where reuse will be measured. The rationale for this strategy is that successful components are often updated with new, improved versions, for example yearly time series charts with information for a new year. To avoid recommendation of old and potentially deprecated components, only fresh reuse information is used. Old components that are still actively reused are not affected by this time-based bias.

The implementation is based on PopRank (Nie et al. 2005), a domain-independent component-level link analysis ranking model. Reuse information is converted into a graph structure where components are linked to components that include them. For example, a table is linked to the slides that contain it. If the component does not have a higher container, it is linked to the users that created or reused the component. The edges of this graph are annotated with the date the reuse took place. The edges are then pruned according to the desired time-frame and the resulting graph is used to calculate the different reuse metrics explained above, based on the incoming edges of a component and the links between components and users.

The metrics are calculated a priori because they are not user or query specific. Results are stored in the repository and are used to rank result lists of components when a user searches for relevant objects, placing components with a high probability at the top of the list. Evaluation results are presented in Section 4.

3. Client Side Applications

Client side applications were developed that enable content upload and component retrieval from within authoring tools. Such applications were developed for Microsoft PowerPoint, Microsoft Word and the Reload Editor. The plug-ins are detailed in the rest of this section.

3.1 Microsoft PowerPoint Plug-in

A plug-in has been developed for Microsoft PowerPoint that enables authors to reuse components stored in the ALOCOM repository from within the application. As shown in Figure 3, a custom Office Task Pane (on the right side) is used for integrating this functionality. This is accomplished with Visual Studio 2005 Tools for the Microsoft Office System (Carter & Lippert 2006).

The plug-in enables authors to search the repository for components they wish to reuse in the presentation they are editing. An author can specify both the component type, such as reference, definition, example, slide, image, or text fragment, and descriptive keywords. Thumbnails of components that satisfy the search criteria are displayed in the ALOCOM Task Pane and metadata associated with a component are shown when the user hovers the mouse pointer over a component in the result list.
The author can incorporate a component into the current presentation by a single mouse-click. The original component is then retrieved and automatically added using built-in copy and paste functions of the PowerPoint API. Doing so, all original content, structure and layout information, including transitions in a slide, is preserved.

![Figure 3. The ALOCOM plug-in for Microsoft PowerPoint [1]](image)

In the opposite direction, authors can add presentations to the repository by clicking the “Save into ALOCOM” button that has been added to the standard PowerPoint menu. When this button is clicked, the presentation is sent to the .Net disaggregation service for decomposition and storage.

### 3.2 Microsoft Word Plug-in

A similar plug-in has been developed for Microsoft Word that automates reuse of Wikipedia components. Definitions, images, references, and text fragments can be individually retrieved. In fact, Wikipedia pages are easy to disaggregate, as the pages have a consistent structure and are stored in well-formed HTML. The first part of the page provides a general definition of the concept. Further content is divided into smaller, clearly labeled, sections.

In contrast to decomposition of presentations, which takes place when they are stored in the ALOCOM server, decomposition of Wikipedia pages is performed upon request at the client side. The Wikipedia search engine is used for retrieval of relevant pages. An HTML parser processes pages that are found and retrieves their content and structure. The first part of the page is retrieved when searching for a definition. Other sections are retrieved when searching for text fragments, disaggregated to the level of single paragraphs, labeled with the title of the subsection to which they belong. Images on Wikipedia have in most cases “alt” attributes that provide a short description. Finally, references are retrieved by parsing the “reference” section in a page.

Similar to the Microsoft PowerPoint plug-in, a custom task pane is defined that enables retrieval of Wikipedia components from within the authoring tool (see Figure 4). A user can specify both keywords and the component type. When searching for text fragments, paragraphs are shown individually, with their surrounding paragraphs. Also the complete section to which the paragraph belongs is retrievable. This support is required, as not many paragraphs are written as standalone pieces of content and often refer to other paragraphs. By showing the surrounding paragraphs, an author is provided with sufficient context to understand the content and chooses whether a paragraph can be reused as a standalone component or should be reused in combination with other paragraphs.
3.3 Reload Editor Plug-in

The key aim of the Reload project is the implementation of a SCORM Content Package and Metadata Editor. The Reload Editor enables users to organize, aggregate and package Learning Objects in SCORM content packages tagged with metadata (Milligan et al. 2005).

A SCORM content package is a self-contained ZIP file. Mandatory Content Package contents are an XML manifest file (imsmanifest.xml), schema definition files referenced by the manifest file and all component files used by the content package.

The manifest file describes the structure and contents of the package. The Reload Editor allows authors to create and edit such manifest file with a convenient graphical interface to visualize the content. A plug-in has been developed for this editor that enables retrieval of components stored in the ALOCOM repository (see Figure 4). A user can enter descriptive keywords and a component type and results are shown in an integrated window. Components in the result list can be dragged and dropped into a specific location in the manifest file. When a component is added, the original component file is retrieved and stored on the client machine. Metadata associated with the component are automatically added to the manifest file.

![Figure 4. The ALOCOM plug-in for MS Word (left) and the Reload Editor (right)](image)

In the opposite direction, support for decomposing SCORM content packages is provided. This decomposition process is straightforward, as components in the content package are already stored individually and described by metadata. The AdvancedContentInserter unpacks the package and stores the components in the repository. Metadata are generated by the AMG framework and merged with metadata that the author might have provided in the manifest file.

4. User Evaluation

The user evaluation assessed the usability and utility of the ALOCOM plug-in for Microsoft PowerPoint. The goals of the evaluation were the following:

- to assess the efficiency and effectiveness of the approach for reusing presentation components;
- to assess the subjective acceptance of the ALOCOM interface;
- to determine to which level of granularity decomposing presentations is relevant;
to assess the efficiency and effectiveness of the ranking algorithm.

4.1 Study Description

A first usability study has been conducted in October 2006 at the K.U. Leuven University. There were 20 participants in the study, which typically results in a reasonably tight confidence interval (Nielsen 2006). Participants were mainly members of the junior staff of the Computer Science Department at K.U. Leuven. Ten participants were members of our own research unit, seven participants were members of the Distributed Systems and Computer Networks research group of the department and three were employed in external organizations.

4.1.1 Tasks

Each participant was asked to create two presentations: one on inheritance and one on exceptions in the programming language Java. The participants were divided into two groups. The first group created the presentation on exceptions in Java without ALOCOM support, and the presentation on inheritance in Java with ALOCOM support. They could use all information available on the World Wide Web for both presentations. The second group did the same, but in a different order. This group created the presentation on inheritance in Java without ALOCOM support, and the presentation on exceptions in Java with ALOCOM support.

The presentation created without ALOCOM support is referred to as \textit{without-alocom presentation} and the presentation created with ALOCOM support as \textit{with-alocom presentation} in the remainder of this paper.

In order to bootstrap the reuse process, 78 presentations on both topics were gathered by a Google search and uploaded to the repository: as described above, they were automatically decomposed and the components were automatically described. In total, 10281 components were made available for reuse, including 2964 slides, 933 images, 6367 text fragments, 12 tables and 5 diagrams.

4.1.2 Data Collection

Camtasia Studio [2] was used to record participant interactions, capturing the screen, voice and webcam video. Participants were also asked to complete a questionnaire after the tasks. The questionnaire was adopted from a usability evaluation of the ARIADNE search tool (Najjar et al. 2005). Rank positions of selected components were logged in Contextualized Attention Metadata files (Wolpers et al. 2007).

4.1.3 Measurements

The following characteristics were measured for the experiment:

- **Time-on-task**: represents the time needed to finish each task. The aim is to investigate whether the use of the ALOCOM plug-in can lead to savings in time. Time is influenced by other factors; however, this comparison is included in order to obtain a first indication of improvements for time-on-task.

- **Manual versus semi-automatic reuse**: the distinction is made between manually reused components and semi-automatically reused components. Manually reused components are components that were added to the presentation by copy-pasting or reproducing existing content, typically found through Google. Semi-automatically reused components are those components that were found and inserted using the ALOCOM plug-in. By measuring and comparing both types of content reuse, a success rate indication of the ALOCOM approach for reusing content is obtained, as authors typically tried the semi-automatic approach first and
inserted content manually if no relevant components were found through the ALOCOM plug-in.

- Component granularity: the granularity of semi-automatically reused component types is measured in order to determine to which level of granularity decomposition of presentations is relevant.
- Satisfaction: user satisfaction was assessed through a questionnaire filled in by each participant after finishing the tasks.

4.2 Results

4.2.1 Time

Table 1 shows the average time participants spent on creating without-alocom and with-alocom presentations. At first sight, the difference is relatively limited: on average, 20.03 minutes were spent creating the without-alocom presentation and 17.79 minutes creating the with-alocom presentation. However, not all participants created presentations similar in length, covered sub-topics or quality in general.

Size normalizations were applied that were adopted from the software quality field (ISO/IEC 9126 1998). A simple normalization that takes into account the number of slides in the presentation shows that on average 3.32 minutes were spent per slide in a without-alocom presentation, whereas 2.2 minutes were spent per slide created with ALOCOM support.

A second normalization was applied that takes into account the number of sub-topics. Some participants created presentations covering many sub-topics, such as polymorphism and dynamic binding for the presentation on inheritance, while others provided only a definition and an example. On average 4.5 minutes were spent on a sub-topic in a without-alocom presentation and 2.9 minutes on a with-alocom presentation sub-topic.

<table>
<thead>
<tr>
<th></th>
<th>without-alocom presentation</th>
<th>with-alocom</th>
<th>Sig.(2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total time</td>
<td>20.03</td>
<td>17.79</td>
<td>0.147</td>
</tr>
<tr>
<td>Time normalized by number of slides</td>
<td>3.32</td>
<td>2.2</td>
<td>0.001</td>
</tr>
<tr>
<td>Time normalized by number of subtopics</td>
<td>4.5</td>
<td>2.9</td>
<td>0.016</td>
</tr>
</tbody>
</table>

To statistically establish whether the difference between these average values is real or a by-product of natural variance, we applied a Paired-Samples T Test. The null hypothesis is that there is no difference between the required creation time for with-alocom and without-alocom presentations. Our alternative hypothesis is that there is indeed a difference. Results were normally distributed. Normality was tested with the Kolmogorov-Smirnov test (Massey 1951).

The null hypothesis can be rejected for normalized time values, as shown in the right column of Table 1 (significance < 0.05). Thus, taking into account the size of presentations, significant time savings are realized when creating presentations with support to automatically reuse existing presentation components. To validate these results, a second evaluation was performed that assessed the quality of the created presentations. This evaluation is presented in section 5.
4.2.2 Reuse in With-Alocom Presentations

With-alocom presentations were further analyzed. The distinction is made between manual reuse, semi-automatic reuse and new components. Manually reused components are components that were added to the presentation by copy-pasting or reproducing existing content, found by a web search. Semi-automatically reused components are those components that were found and inserted using the ALOCOM plug-in. New components represent content the participant created from scratch, without using an existing resource.

Figure 5 (left) shows reuse patterns of individual participants. Some participants reused about the same amount of components manually than semi-automatically. Also, the amount of new components is high for some participants (more than 40%). Few participants created presentations without manual reuse.

![Reuse patterns of individual participants](image1)

![Reuse/component type](image2)

Figure 5. Reuse patterns of individual participants (left); reuse/component type (right)

Table 2 shows that on average 57% of presentation components were reused semi-automatically using the ALOCOM plug-in. 18% of the components were reused manually, whereas 25% are new components. There is no significant difference if we compare this data for the presentation on exceptions in Java and the presentation on inheritance in Java, although more components were available covering topics on inheritance. The values were normally distributed and compared with a Paired-Samples T Test.

<table>
<thead>
<tr>
<th></th>
<th>Manual</th>
<th>Semi-automatic</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>0.18</td>
<td>0.57</td>
<td>0.25</td>
</tr>
<tr>
<td>Presentation on inheritance (1)</td>
<td>0.19</td>
<td>0.58</td>
<td>0.23</td>
</tr>
<tr>
<td>Presentation on exceptions (2)</td>
<td>0.18</td>
<td>0.55</td>
<td>0.27</td>
</tr>
<tr>
<td>Comparing means (1) and (2)</td>
<td>0.737</td>
<td>0.121</td>
<td>0.791</td>
</tr>
</tbody>
</table>

Comparing manual and semi-automatic reuse, we see that 76% of reused components were reused semi-automatically, whereas 24% were reused by copy-paste actions or reproduction of content. These values are a success rate indicator of the ALOCOM approach for reusing content, as participants typically tried the semi-automatic approach first and inserted content manually if no relevant components were found through the ALOCOM plug-in.
4.2.3 Granularity

Figure 5 (right) shows the reuse rate for semi-automatically reused component types. Complete slides are most often reused, probably because many slides represent a single idea or topic and are thus easy to reuse in a new context. The reuse of text fragments is also significant. This is an interesting result, as it illustrates that breaking content down to the level of a single text fragment is useful. Images were not frequently reused; however, this result is probably influenced by the topic of the presentations.

4.2.4 Effectiveness of the Ranking Algorithm

As illustrated in Figure 6 (left), 26% of selected components were presented in the top position of the result list, 77% in the top 5 and 87% in the top 10.

This result is consistent with previous findings (Joachims 2002, Ochoa & Duval 2007) that indicate that popularity of a document is highly correlated with its relevance for a closely related group of users, in this case computer scientists from the same department, performing a similar task. Also interesting, a non-negligible amount of components (10%) were retrieved from positions between 15 and 52. This result goes against the accepted belief that users select only results in top positions. This can be explained as the experimental setup interfering with the normal behavior of users.

![Figure 6. Effectiveness of the ranking (left) rank versus popularity (right)](image)

The most reused components tend to appear in the first positions in the ranking. As shown in Figure 6 (right), the rank position of the most popular components (6, 5 or 4 reuses) is inside the top 5. This result implies that if the ranking is refined with the new reuse information, the position of the elements in the list will not change abruptly. This degree of stability is a desirable behavior for a ranking algorithm. When the order of the results would be volatile, the trust of users in the system could decrease.

4.2.5 Findings and Recommendation.

In this section, findings and recommendations of the participants are discussed.

4.2.5.1 Lack of Context

Some participants remarked that more context is required for successful content reuse. They want to be able to retrieve the next and previous slide for a specific slide in the result list, or even the
complete presentation(s) to which the slide belonged. Similar support is needed for other component types.

4.2.5.2 Behavior Change
It was noted that this way of reusing content requires a behavior change, as it is different from the usual practice of copy-pasting or reproducing content. It was reported that savings in time would be remarkable; however, a period of adaptation is required.

4.2.5.3 Drag and Drop Support
Many participants expected drag and drop support for inserting components. There is click-support for inserting a component: clicking a component in the result list will insert the component at the currently selected location. However, it is not possible to drag the component to a different location in the presentation, due to limitations of the PowerPoint API.

4.2.5.4 Garbage Content
Not all components are reusable. As components are created by decomposing existing content automatically, it was expected that not all components are valuable for reuse. Results are ranked according to the number of times a component is reused. Hence, the impact of this issue will decrease over time.

4.2.5.5 Less Consistent Layout
Some participants noted that it is hard to keep the layout of different components consistent. The layout of slides is automatically adapted to the template the author is using. However, if the author changed, for instance, the font color of a text fragment in one particular slide, this modification is preserved when reusing the slide. Although desirable in some cases, this was reported as a difficulty.

4.2.5.6 More Valuable for Reuse of Own Content
Participants remarked that the use of the ALOCOM plug-in would be most valuable for reusing their own presentations.

4.2.6 Overall Satisfaction
Table 3 presents the responses of participants to questions concerning the overall use of the ALOCOM plug-in. The questionnaire was adopted from a usability evaluation of the ARIADNE search tool (Najjar et al. 2005). The popular attitude scale with seven points (ranging from 1 - poor to 7 - good) was used to measure the response of participants on the overall use of the plug-in.

<table>
<thead>
<tr>
<th>Table 3. Satisfaction</th>
<th>mean (ranging from 1–7)</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ease of use</td>
<td>6.15</td>
<td>0.69</td>
</tr>
<tr>
<td>Information organization</td>
<td>5.23</td>
<td>0.93</td>
</tr>
<tr>
<td>Use of terminology</td>
<td>4.92</td>
<td>1.5</td>
</tr>
<tr>
<td>Navigation</td>
<td>6.07</td>
<td>1.04</td>
</tr>
<tr>
<td>Search and reuse of components</td>
<td>5.69</td>
<td>1.49</td>
</tr>
<tr>
<td>Result list easy to read</td>
<td>4.92</td>
<td>1.5</td>
</tr>
</tbody>
</table>
The mean for the level of ease-of-use was more than 6, meaning that the participants found the ALOCOM plug-in easy to use. The level of information organization and search and reuse of content components was perceived as moderate (mean 5.23 and 5.69 respectively). We believe that this is related to the fact that there is a lack of context (it is not possible to automatically retrieve the original component to which a component belonged) and the fact that there is no drag and drop support.

Result lists were found rather difficult to read (mean 4.92). This result is a consequence of the fact that preview thumbnails of slides containing much content are difficult to read. We have worked on a solution that enables users to enlarge individual components. Each component in the result list has a context menu item that provides this functionality. This solution will resolve the issue if only few components are difficult to read.

5. Quality Evaluation

In a follow-up evaluation, the quality of with-alocom and without-alocom presentations was assessed by a group of 19 participants. This evaluation was necessary for obtaining a more accurate estimation of the effectiveness and efficiency of the ALOCOM approach for reusing presentations.

Following a common practice to reduce subjectivity in a quality evaluation, an evaluation framework was used. In (Knight & Burn 2005), an overview is provided of the most common dimensions of Content Quality frameworks. Four dimensions that were relevant in the context of the experiment were used to evaluate the quality of the presentations: accuracy, completeness, relevancy and conciseness.

In an accurate presentation, the content contained in the presentation is correct, reliable and free of error. Completeness is defined as the extent to which information is not missing and is of sufficient breadth and depth for the task at hand. Relevancy measures whether the content contained in the presentation is applicable and helpful for the task at hand. Finally, in a concise presentation, content is broken up into smaller chunks that can be easily shared with an audience.

Participants in the experiment were requested to read the definition of each parameter before grading the presentations. The definitions were also available during the evaluation process.

The experiment was carried out online using a web application. After logging in, the system presented users with instructions. After reading the instructions, users were presented with a list of 20 randomly selected presentations. Once users had reviewed a presentation, they were asked to give grades on a 7-point scale, from “Extremely low quality” to “Extremely high quality”, for each parameter. Only participants that graded all presentations were considered in the experiment.

The experiment was available for 2 weeks. During that period, 24 participants entered the system, but only 19 completed the evaluation. From those 19 participants, 13 were postgraduate students, 1 had a Ph.D. degree and 5 were active in software development. All participants had a degree in computer science.

5.1 Data Analysis

Because of the inherent subjectivity in measuring quality, the first step in the analysis of the data is to estimate the reliability of the evaluation. In this kind of experiment, the evaluation is considered reliable if the variability between the grades given by different reviewers to a particular presentation is significantly smaller than the variability between the average grades given to different presentations. To estimate this difference, we used the Intra-Class Correlation (ICC) coefficient (Shrout and Fleiss 1979), which is commonly used to measure the inter-rater reliability. We
calculated the average ICC measure using the two-way mixed model, given that all reviewers grade the same sample of presentations. In this configuration, the ICC is equivalent to another widely used reliability measure, the Cronbachs alpha (Cronbach 1951). The results for each quality parameter are reported in Table 4.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ICC (average, two-way mixed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completeness</td>
<td>0.927</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.766</td>
</tr>
<tr>
<td>Conciseness</td>
<td>0.881</td>
</tr>
<tr>
<td>Relevancy</td>
<td>0.837</td>
</tr>
</tbody>
</table>

Generally, ICC values above 0.75 indicate good reliability between measures. None of the values fall below this cut-off value. Hence, the ICC suggests that reviewers provided similar values and further statistical analysis can be performed.

The second step is to assess whether there is a difference between the average grade given to *with-alocom* presentations and the average grade given to *without-alocom* presentations. These average values are presented in Figure 7. To statistically establish whether the difference between average values is real or a by-product of natural variance, we applied a Paired-Samples T Test. Our null hypothesis is that there is no difference between the grades given to *with-alocom* and *without-alocom* presentations. Our alternative hypothesis is that there is indeed a difference. The results are presented in Table 5. Results were normally distributed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>T-value</th>
<th>Significance (2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completeness</td>
<td>-8.094</td>
<td>0.0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>-1.412</td>
<td>0.160</td>
</tr>
<tr>
<td>Conciseness</td>
<td>-4.352</td>
<td>0.0</td>
</tr>
<tr>
<td>Relevancy</td>
<td>-2.981</td>
<td>0.003</td>
</tr>
</tbody>
</table>

The null hypothesis can be rejected for most of the parameters (completeness, conciseness and relevancy). The significant difference found in the completeness parameter indicates that users were
able to create more complete presentations when provided with support to reuse presentation components. The significant difference found in the conciseness parameter indicates that content extracted from existing presentations is more suitable for reuse as it is already presented in a form that can be shared with an audience. Furthermore, users were able to find more relevant content for with-alocom presentations. No significant difference was found in the accuracy parameter. As the presentations were created by members of the junior staff of the Computer Science Department at K.U. Leuven, it was expected that no major mistakes would be made in creating presentations on inheritance and exceptions in Java.

6. Discussion

Although no direct saving in time was perceived, results of the quality evaluation indicate that providing on-the-fly access to presentation components in an authoring process enhances the quality of presentations. Presentations created with ALOCOM support are significantly more complete, concise and relevant. The results in completeness are consistent with the size normalizations applied to time values. Hence, there is also a significant improvement in time savings.

Results of the user evaluation indicate that the plug-in can be used in a successful way: 76% of reused components were reused semi-automatically. However, usability issues need to be resolved in order to make this kind of content reuse more efficient. Most important is the context issue. The user interface should be extended with the functionality to retrieve the component to which a component in the result list originally belonged. Furthermore, it is important to enable navigation in the original structure of presentations. For instance, support is needed to retrieve the next and previous slide for a specific slide in the result list.

The consistent layout issue cannot be improved, as built-in copy and paste functions of the PowerPoint API are used for adding an existent slide to a presentation. If a user would manually copy-paste a slide, the same problem with consistency arises. Drag and drop support is also difficult to integrate.

The method used can be classified as a “discount usability engineering” approach (Nielsen 1989), as it is definitely not "the perfect" method for evaluation and will not give absolute results. However, it enabled us to obtain a good indication of improvements towards savings in time or enhancements of quality and to highlight usability issues. To generalize the findings, a second evaluation should be performed with a group of participants from outside the computer science field.

The evaluation of the ranking algorithm indicates that ranking based on aggregation relationships provides an easy and effective way to rank the results when the components in the repository are covering similar topics. Nonetheless, a small, but non-neglectable, amount of components were not ranked according to their relevance to the users in the experiment. These errors could be solved when more usage data is fed into the popularity ranking algorithm or using more advanced techniques, for instance personalized or contextual relevance ranking (Ochoa & Duval 2007), during the ranking calculation.

To enable such mechanism, the Contextualized Attention Metadata (CAM) framework (Wolpers et al. 2007) can be used for capturing the attention a user spends on content. ALOCOM client applications already generate such streams that capture activities within the application with timestamp and content-related data. For instance, the time a user spends working on a document, the queries that are performed and the components that are reused by the user, are captured. This data can be used for building user attention profiles that represent actual interests of users based on content they worked with. The use of such profiles enables personalized ranking.
7. Related Work

In recent years, a lot of research has been dedicated to develop flexible documents that are generated by assembling smaller, reusable, components. However, few approaches support automatic decomposition of existing documents. Instead, guidelines are often provided to decompose content manually or to create new components suitable for reuse. Such guidelines are for instance described in the dLCMS project (Schluep 2005). Some commercial content management systems, such as Vasont [5], also use a manual transformation process to support content reuse. The approach presented in this paper is more scalable, as it attempts to automate content reuse for pre-existing documents.

MagIR (Kienreich et al. 2005) is a system that supports automatic content transformations. Like in ALOCOM, these transformations are supported for PowerPoint presentations and include content decomposition. Decomposition is supported to the level of slides only, while ALOCOM also extracts smaller components, such as tables, diagrams, images and text fragments. Results of the user evaluation presented in this paper indicate that these fine-grained components are also often reused. MagIR is used for creation, administration and reutilization of PowerPoint slides in a corporate context and is aimed at reducing storage costs. The system has been evaluated in that context and results indicate that storage costs are significantly reduced.

Slide executive [4] is a commercial product that also supports reutilization of Microsoft PowerPoint slides. Individual slides can be retrieved in a browser and dragged and dropped in a PowerPoint presentation. Like MagIR, decomposition is supported to the level of slides. Add-ins are provided to export PowerPoint slides to images in different formats and to import multiple images at once. However, no tight integration for component searching from within the application is supported. No information has been found whether the system has been evaluated.

The TRIAL-SOLUTION project is developing tools to create and deliver personalized teaching materials that are composed from a library of existing documents on mathematics at undergraduate level (Lenski & Wette-Roch 2001). The focus of the project is on document (de-)composition and exchange for reuse. The TRIAL-SOLUTION System contains a splitter that decomposes document source files into a hierarchy of slices. For this decomposition, the presentation style of a particular author is taken into account. Also, it takes care of counters and key phrases assigned by the author. In addition, decomposed content is manually revised. The main difference is that the methodology for decomposing content is semi-automatic and therefore less scalable.

The Legacy Document Conversion (LegDoC) project is offering advanced techniques to automate conversion of legacy documents to XML (Lecerf & Chidlovskii 2006). Layout-oriented formats like PDF, PS and HTML are automatically converted to semantic-oriented annotations. The table of contents of a document is used as a basis for document structuring. The approach is promising, as no assumptions are made about the structure of source documents. Integrating such approach into the ALOCOM framework would enable structuring and decomposition of unstructured or semi-structured documents that contain a table of contents.

8. Conclusions

In this paper, a framework has been presented that enables scalable content reuse by decomposing presentations, Wikipedia pages and SCORM content packages into reusable components and supporting their on-the-fly reuse in mainstream authoring tools. The analysis of the results of the user and quality evaluations of the approach for presentations indicates that there is a significant improvement of the quality of presentations and a significant time saving benefit.
The successful application of the approach largely depends on the performance of all its components. An accurate reuse detection mechanism is required for avoiding duplicates, components need to be precisely described to enable their retrieval and a ranking mechanism is a key requirement when dealing with a large number of components. In the next steps, a personalized ranking mechanism will be integrated that is expected to further improve the efficiency of the approach. Techniques for transforming unstructured or semi-structured documents need to be investigated to enable component reuse for a wider variety of documents. Finally, evaluations of the MS Word and Reload plug-ins are required to assess whether these approaches result in a similar impact on effective and efficient content reuse.

Acknowledgements

We thank professor M.-F. Moens for her useful comments. This research is supported by the K.U.Leuven research council through the BALO project, the Interdisciplinary Institute for Broadband Technology (IBBT) through the Acknowledge project, and the European Commission through the ProLearn Network of Excellence on Professional Learning [3].

References


Notes