Future Internet **2010**, *2*, 645-661; doi:10.3390/fi2040645

OPEN ACCESS

future internet

ISSN 1999-5903

www.mdpi.com/journal/futureinternet

Article

Simplifying the Scientific Writing and Review Process with SciFlow

Frederik Eichler and Wolfgang Reinhardt *

Computer Science Education Group, University of Paderborn, Fuerstenallee 11, 33102 Paderborn, Germany; E-Mail: mail@frederik-eichler.de

* Author to whom correspondence should be addressed; E-Mail: wolle@upb.de; Tel.: +49-5251-60-6603; Fax: +49-5251-60-6336.

Received: 13 September 2010; in revised form: 30 November 2010 / Accepted: 2 December 2010 / Published: 6 December 2010

Abstract: Scientific writing is an essential part of a student's and researcher's everyday life. In this paper we investigate the particularities of scientific writing and explore the features and limitations of existing tools for scientific writing. Deriving from this analysis and an online survey of the scientific writing processes of students and researchers at the University of Paderborn, we identify key principles to simplify scientific writing and reviewing. Finally, we introduce a novel approach to support scientific writing with a tool called SciFlow that builds on these principles and state of the art technologies like cloud computing.

Keywords: scientific writing; survey; word processors; cloud computing

1. Introduction

Scientific writing is an essential part of a student's and researcher's life. Depending on the particular field of study, papers have to be written and written assignments have to be handed in. As the end of studies approaches, most students are asked to prove their ability to work in a scientific manner by writing a thesis. Moreover, Ph.D. students, research assistants and other scientific staff publish their research results in conference or journal articles. Depending on the type of the written artifact their work undergoes review that rates the quality, novelty or structure of those texts. Very often writers are limited to a specific template using a word processors of their choice.

In this paper we introduce a novel approach to support scientific writing with a tool called SciFlow. The principles SciFlow was built on were derived from the results of an online survey concerning the scientific writing processes of students and researchers at the University of Paderborn. During the design phase we have identified key principles to support the writing process and then built a concept around those principles. Furthermore, we focused on the fact that scientific documents are governed by a pre-existing set of rules and regulations like spacing, citation style, and referencing. Using the previously mentioned principles we will describe the development and implementation of a concept that uses web services and cloud computing to provide a seamless writing experience to a user. Before, we will describe scientific writing and the various available word processors.

2. The Particularities of Scientific Writing

In this section we will present specific roles and introduce the phases of scientific writing, examine existing tools and word processors that support the writing process and discuss their usefulness for scientific writing.

2.1. Roles and Phases of Scientific Writing

The scientific writing process usually involves multiple people and multiple structured phases of activities. [1] set up a number of roles and phases for scientific writing. The identified roles are the *writer*, the *consultant*, the *editor*, and the *reviewer*. The consultant actively participates in the different phases of writing but does not compose text himself, whereas the editor actively corrects text. This is also the distinction to the reviewer, who only provides comments on a document. For writing itself, multiple strategies are possible. A *single writer* composes a document with minimal assistance from others. In a *scribe session* one individual records the group's discussions and *separate* or *joint writers* contribute to a document to equal parts [1].

Functionality like commenting is not only influenced by the different roles, but also by different phases in scientific writing. The phases or activities that [1] identifies are: (1) brainstorming, (2) research, (3) planing, (4) writing, (5) editing, and (6) reviewing. During the brainstorming phase authors generate ideas for the document, followed by the researching phase where information is gathered from sources outside the writing group. During the planning activities the authors create a first outline for the document and divide work packages and responsibilities amongst each other. When writing, the authors' ideas are transformed into text and graphical representations, followed by making changes to the text and outline (editing phase). In the final phase the authors gather comments about the document from internal and external reviewers. Whereas the writing processes differs between a single writer and a collaborative writing group, the phases of scientific writing stay the same for each writing type.

All of the phases can be supported with the help of special-purpose tools or all-in-one writing suites. In the following section we introduce the specifics of classic word processors and modern tools for scientific writing.

2.2. *Classic Word Processors and Tools for Scientific Writing*

As a popular word processor, Microsoft Word supports a great deal of functionality, which is suitable for scientific writing [2]. It allows for the creation of a table of contents and the insertion of figures [3]. Furthermore, Microsoft Word can be used to comment on documents and thus may also be used for the review of documents [4]. Microsoft Word is easy to use and known by most people involved in scientific writing (cf. Section 2.3). Tools like Citavi [5] and Zotero [6] directly integrate with Microsoft Word and allow to manage citations and online references. Zotero also integrates with Google Scholar [7], a search portal tailored to scientific articles and other comparable documents [8]. Zotero is also available for Open Office [9], which offers similar features like Microsoft Word, but is developed as Open Source [10].

Apart from traditional word processors, existing literature also proposes the use of $\LaTeX$ for writing, especially within the technical field [11]. $\LaTeX$ is a typesetting system and markup language designed to publish technical and scientific documentation. It is available as free software and is supported by various writing environments. These may be preferred by some people, since $\LaTeX$ itself does not provide *What You See Is What You Get* (WYSIWYG) functionality, as Microsoft Word or Open Office do. Rather than providing a fully laid out document, a $\LaTeX$ document consists of text, commands and document markup. This causes for a separation of content and layout and has also be described as *What You See Is What You Want*, acknowledging that the edited document is only a representation of the document contents. It is not an exact replica of the final layout [12]. $\LaTeX$ allows for the creation of DVI, PS or PDF [13] files that fulfill the requirements of scientific and technical documents (e.g. table of contents, figures, mathematical equations). Additionally, the integration of BiBTeX files allows a writer of $\LaTeX$ documents to maintain and style a bibliography individually [14] .

Apart from these desktop applications, the introduction of asynchronous data transfer from the browser to an entity in the web (e.g., through AJAX) allows for real-time collaboration online. A prominent example for this is Google Docs [15], which does not only allow for real time editing of documents, but also tracks revisions of the different contributors. However, Google Docs itself does not provide support for the earlier mentioned roles, nor phases, natively [16]. Scientific writing with Google Docs must be based on conventions [17], e.g., how to format a document, rather than through pre-defined settings. An example for such a convention is the formatting of running titles in page headers, which can not be automatically derived from the chapter titles. Additionally, Google Docs lacks support for managing figures or references [16]. Adobe Buzzword [18], an online word processor based on Flash technology, does support multiple roles (*i.e.*, author, co-author, reviewer and reader). Nevertheless, for the time being it also lacks the features that are necessary in scientific writing, like the support for a bibliography and referencing parts of a document.

2.3. *Results of a Survey on Scientific Writing*

In order to gain a deeper understanding of how scientific writing is accomplished, which tools are being used and how authors rate the strengths and weaknesses of the tools they are using, we carried out a survey (N = 1,603) amongst students and researchers from the University of Paderborn in September 2009. The survey focused on six main parts: (1) document type and software used,

(2) general software features, (3) revision of documents, (4) backup of documents, (5) restrictions in writing (e.g., through templates), and (6) features that were liked or disliked in the software used.

As for general results, Microsoft Word was used by 67.1% (902 people), LaTeX by 18.3% (246 people), Open Office was used by 11.1% (149 people) and other tools were used by 3.6% (48 people, N = 1,345). Out of these participants, 39.5% (468 people) answered the survey after having written a thesis and 54,5% (646 people) related the survey to having written a homework assignment. The remainder had written a scientific paper (4.3%/51 people), a dissertation (1.8%/21 people) or other scientific documents (71 people, N = 1,258). In the following we only present the specific results for Microsoft Word and LaTeX, reflecting their high percentages of use. For neither of them we regard different versions of the software. Most of the questions participants were presented a six-point Likert scale ranging from 1 to 6.

Regarding general software features, we found out that participants felt well supported by either tool. Specific averages for the questions range from 4.73 for *PDF export* to 4.53 for the *creation of a table of contents, formatting and page numbers*. They result to 4.31 for the *initial software setup* down to the lowest average of 4.19 for *citing and referencing*. In direct comparison, Microsoft Word outperformed LaTeX for the *easier setup* but lost on all other parts of this section. The participants found LaTeX to be 23% easier to use for citations and referencing, 31% better for the *creation of the table of contents, page numbers and general formatting* and 25% better for the *creation of a printable document*.

The question towards how well the software supported the user in revising the document consisted of four parts. *Sharing the document* was the best supported feature (average: 5.0) and *revising the document* on one's own ranged second with 4.64. *Achieving a desired layout* and *merging changes from others* averaged at 4.28 and 4.20 respectively. The direct comparison between Word and LaTeX shows a marginal 0.02 and 0.14 point lead for Word on merging other's work and revising. As for the document layout, LaTeX achieves 19% (0.94 points) better results on average and gains 5% (0.21 points) on *sharing the document*.

Block four of the survey was dedicated to finding out how participants performed their document backups. The survey shows that more than half of the participants did manual backups on USB/DVD (856 people) and another 29.8% (352 people) used hard drive backup tools to back up their work. Online backup and version control resulted to less than 10% in each case (N = 1,181, multiple answers were allowed). As for the frequency of backups, 82.2% (705 people) performed backups rather often or very often. 89.8% (1,018 people) had an internet connection *often or always*. Only 1.2% had no or only little access to the Internet (N = 1,134).

Finally, we asked participants which features they liked about the software they were using (677 statements) and which ones they did not like or missed (613 statements). However, we did not pose these questions to find out about the strengths and weaknesses of specific software, but rather to explore which features of a writing software are essential to the participants. The statements were very subjective, contained more than one piece of information and so we clustered them in categories according to their message.

For Microsoft Word the most frequent categories were *easy to use*, *I am using it, because I am used to it* and *a lot of people use it*. The important distinction between *easy to use* and *I am using it, because I am used to it* is that the latter not necessarily means that the user liked the software, but rather that this was the only software he knew. Word users disliked that maintaining a table of contents was difficult (even with use of the feature provided by Word), that formatting often did not work as expected and that the footnotes and page numbers (including different numbers for the first pages of a document) were hard to set up.

For LaTeX, the participants stated that it generates professional layouts, lets the user focus on writing in stead of the layout, and that the table of content generation works well. Generating professional layouts here means that it was easy to create a document that followed the given restrictions for scientific writing. The negative statements for LaTeX were lead by the fact that participants perceived it as being difficult to position tables and figures. In addition they did not like the lack of WYSIWYG features in the particular LaTeX editor they used and that sharing the document for review was hard.

2.4. *Requirements for a Scientific Word Processor*

[12] identifies criteria to measure the quality of a graphical text editor with regard to scientific writing. These are *user friendliness*, *simplicity*, *controllability*, and *(the lack of) distraction and transparency* amongst others. As part of an editor's feature set, [12] demands the existence of structural markup, thus for the editor to display the logical structure of a document alongside the layout. The author further expects a distinct set of primitives (e.g., chapters, sections, *etc.*) and macros to make additions to the markup possible. Moreover, the author demands that the typesetter shall be fast and the user interface shall be responsive. Lastly, it proposes the requirement that user input (e.g., a mathematical equation) should be transparent in terms of the generated output.

Taking the results from our survey into account we explored that scientific writing is bound by a number of restrictions: the survey reveals that 67.8% (763 people) were *restricted by font size and spacing* and another 69.2% (678 people) were *bound by a specific citation style*. 28.3% (319 people) were handed a *document template* and only 20% (225 people) were given no restrictions at all (N = 1,126, multiple answers were allowed).

A scientific word processor thus should be (1) user friendly, (2) simple to use, (3) easy to control, (4) non-distractive, (5) transparent, (6) able to follow given templates, (7) allow multiple citation styles, (8) supportive with regards to recurring writing phases, and (9) able to support different roles.

2.5. *Limitations of Existing Concepts*

Existing concepts for scientific writing cover a great deal of tasks a writer is confronted with [19,20]. So far, we put programs like Microsoft Word into the context of scientific writing and introduced LaTeX as well as collaborative concepts to compose scientific documents. Even though all of these tools make scientific writing possible, they also have limitations. Our survey revealed that mostly Microsoft Word, LaTeX and Google Docs were used for scientific writing. After having introduced their advantages in the previous sections, we will now name major limitations in respect to scientific writing for each of these tools.

Google Docs, for instance, offers no way to use referencing software directly [16]. While the possibility to work on the same document in real time may be useful in some situations, in a scientific writing setting, the lack of support for reference management and figures is a serious limitation.

Moreover, to be suitable for scientific writing in general, the support of various citation styles should be possible. Furthermore, it is not possible to give a person the ability to comment on a document without giving them full access to text editing. This may not be desirable in a review situation.

Microsoft Word allows scientific writing through built in functions or plugins like Citavi. However, even with the mature commenting mechanism that Word offers [3], the user still has to merge the different versions by hand. Once a document has been handed to colleagues for review, the author will end up with several versions of the document. Possibly, he has done his own modifications since the beginning of the review, which then tasks him to merge his new version with the comments colleagues have made. Additionally, [4] found that the effort for reviewers to start a review with Word was high (as opposed to an interactive review system). Lastly, Microsoft Word may become impractical for long documents, as [21] notes.

LaTeX shares the limitation of review with Word. While it is possible to send PDF or even plain LaTeX documents out for review, a manual merging process has to be employed, to reduce the various review versions to one document once review is completed. Secondly, LaTeX is perceived to have a steep learning curve [22], which may prevent users from choosing it for writing. This may be true especially those writers, who do not come from a technical field. In this respect [23] notes that, while LaTeX gives the author freedom to set a document in the way he likes, it also demands for the author to have an idea about the nature of typographical composing.

The use of Subversion for backups has limitations itself. For two people to review and edit the same document they have to obey by certain rules. For instance, they have to turn off automatic line wrapping, not to cause a revision, when no real changes to the content were made. While it may be possible to obey by these rules, they rely on oral or written conventions, which have to be learned by anyone who wants to participate in collaboration or a review.

There may be ways to enhance Microsoft Word and LaTeX with external tools and plugins in a way that most limitations become void, but a user who is new to these tools will have to discover them during a long lasting process and install a working solution by hand. In the following section we introduce a novel software approach to scientific writing which is derived from the results of our survey and the limitations of existing software. Moreover, it was designed to let writers focus on writing itself, not on dealing with a word processor to set everything up.

3. The SciFlow Approach to Scientific Writing

It is the specific goal of the SciFlow concept to simplify the scientific writing and review process. The concept is based on two basic assumptions. (1) Scientific writing is a distinct subset of writing grouping recurring activities like research, writing, referencing and reviewing (see Section 2.1). (2) Furthermore, groups of authors (e.g., participants of a conference) have to obey by the same layout rules or templates (see Sections 2.3 and 2.4). In this paper we will focus our efforts on the phases (3) planing, (4) writing, (5) editing, and (6) reviewing, as introduced in Section 2.1.

3.1. Granularity of the Problem

Simplifying the scientific writing and review process is a broad field. To specify the problem more exactly, we have reduced the general problem to four primary issues. The concept has to concern itself with a method of (1) **input**, meaning it has to provide a word processor that can structure text and manage figures as well as references (planing, writing and editing phase). Moreover, it has to offer a method for (2) **review** when an author decides to share his work as a PDF, a printout or a website (review phase). This review has to be facilitated in a manner that does not require the author to manually merge an annotated document with his version of the document.

A concept that simplifies the scientific writing process shall furthermore include a component that is capable of creating a professional (3) **output**. This output has to obey the rules that have previously been setup in a template. The fourth primary issue is (4) **flexibility**. This means that the concept has to be able to *scale* (e.g., document size, number of users) as well as be *portable* and *maintainable*, so it may be updated for specific tasks related to scientific writing (e.g., support for mathematical equations).

In the following sections we will present a writing client, a document review website, as well as components, that are connected to the writing client through web services. All of these components make use of a scalable communication infrastructure.

3.2. Four Guiding Principles

To solve the previously discussed issues, SciFlow uses four principles. We have deducted these principles from the survey, as discussed in Section 2.3 (also see [24]). The principles are (1) **scalability**, (2) **simplicity and transparency**, (3) **safety and recoverability** and (4) **ease of revision**. This section will introduce the principles by explaining how they form the SciFlow concept. Each of these explanations will be followed by a description of how the concept has been implemented as part of the first SciFlow prototype.

3.2.1. Scalability

The SciFlow concept is based on a distributed cloud architecture, using web services (e.g., for PDF creation, backups or review) to outsource what does not have to be computed on the author's machine. For each of these web services multiple instances may be launched to cope with high work load. While the distributed structure keeps the SciFlow writing client slim and quick to install, it also raises the question of scalability; *i.e.*, the constant availability to all users.

SciFlow uses a service directory to accomplish loose coupling between all connected services. This directory holds references to all instances a web service. For the directory not to become a bottleneck itself, it is hosted on a distributed system, ensuring high availability (e.g., Amazon Web Services [25]).

While distributing the directory onto several machines may be sufficient to implement a simple service directory, more complicated web services require a greater effort to scale. It is vital to the scalability of the concept that three conditions hold true.

1. When the work load a web service experiences, depends on the number of people who use the system, it has to be possible to *instantiate this service an arbitrary number of times* without manual

intervention. An example for such a service would be the document publishing service, which creates PDF files from a document and a given document template. In this case, the demand for the service will rise with the number of authors, thus more instances will have to be made available.

2. To manage an arbitrary number of services, communication between components, e.g., the client and the services, has to be robust. This calls for an asynchronous communication structure, so messages may be delivered when a service is available and held back when it is unresponsive. Such a communication structure will also allow to handle multiple instances of a service: Addressing a message to an arbitrary service allows for service instances with free capacities to accept the message, instead of letting the message sender choose the instance. If no such service is available, the message will wait until the situation changes or more instances of the particular service are launched. This calls for monitoring as we shall now explain.
3. To ensure scalability, the number of services within the SciFlow system has to reflect the number of requests, made to a particular service. For the asynchronous communication structure this means that the number of messages requesting a specific service has to be monitored. Since (1) all services can be launched an arbitrary number of times and (2) the asynchronous communication structure allows for launching and stopping of services without interrupting ongoing operations, a high number of messages may be dealt with by launching an appropriate amount of new services.

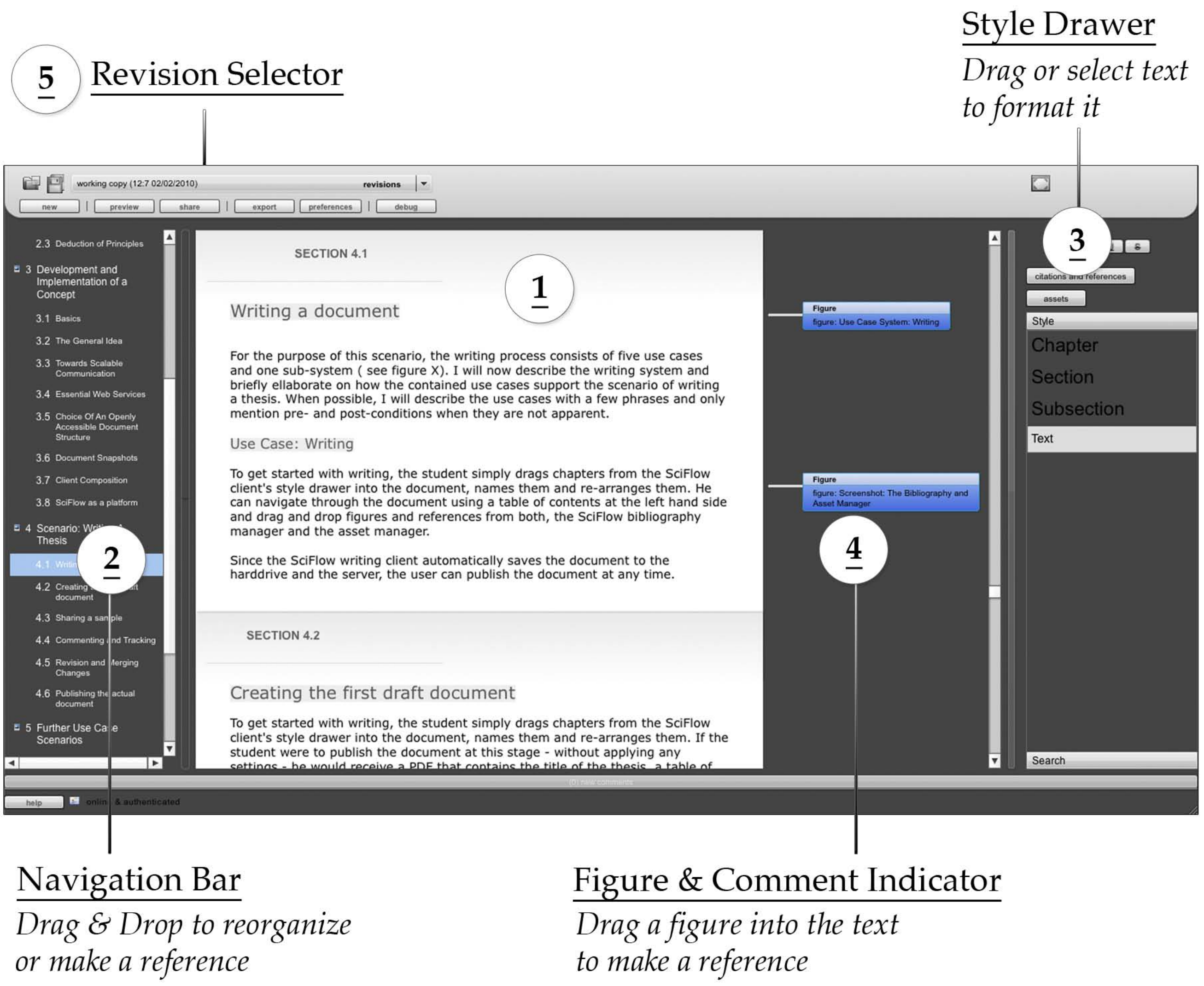
IMPLEMENTATION To honor the three conditions we have just introduced, each component of SciFlow has been implemented to be started autonomously. Additionally, services like the document publishing service may be run on most linux environments with the *dblatex* packages and *DocBook XSL templates* templates installed. This makes the use of cloud computing possible (e.g., Amazon EC2).

With *Amazon Simple Queue Service (SQS)* we have found an implementation of an asynchronous message structure, that was built with high load in mind [26]. SQS allows us to monitor the number of messages at any given moment [26] and to also handle a high number of read and write requests to the message queue. SQS accomplishes this by sacrificing data consistency. It is run on a distributed system, meaning that there are multiple instances of the message queue that synchronize themselves. While this may cause for two simultaneous read requests to differ from each other, all instances of the message queue will be consistent eventually. For SciFlow, this is sufficient.

3.2.2. Simplicity and Transparency

The writing client is the SciFlow component with the greatest exposure to the user. Thus, it is vital that the client is easy to use and transparent in its interaction with the user (see Section 2.4). However, the set of actions it is required to fulfill is limited, thus it may be kept simple. The writing client has to enable the author to structure text (*i.e.*, paragraphs, chapters, footnotes, *etc.*) and maintain a list of figures and references. What it does not have to do is to provide extensive layout and formatting settings. In general, the author will have an only marginal influence on how a document has to be laid out or how citations are formatted. It stands to reason that document setup is not a part of the writing client but is set up by the entity that provides the software (e.g., a conference, a professorship or a university department).

Figure 1. SciFlow writing client.


5
Revision Selector
Style Drawer
Drag or select text to format it
working copy (12:7 02/02/2010)
revisions
new
preview
share
export
preferences
debug
2.3 Deduction of Principles
3 Development and Implementation of a Concept
3.1 Basics
3.2 The General Idea
3.3 Towards Scalable Communication
3.4 Essential Web Services
3.5 Choice Of An Openly Accessible Document Structure
3.6 Document Snapshots
3.7 Client Composition
3.8 SciFlow as a platform
4 Scenario: Writing a Thesis
4.1 Writing
4.2 Creating a draft document
4.3 Sharing a sample
4.4 Commenting and Tracking
4.5 Revision and Merging Changes
4.6 Publishing the actual document
5 Further Use Case Scenarios
1
2
3
4
Figure
figure: Use Case System: Writing
Figure
figure: Screenshot: The Bibliography and Asset Manager
citations and references
assets
Style
Chapter
Section
Subsection
Text
Search
(0) new comments
help
online & authenticated
Navigation Bar
Drag & Drop to reorganize or make a reference
Figure & Comment Indicator
Drag a figure into the text to make a reference

SECTION 4.1

Writing a document

For the purpose of this scenario, the writing process consists of five use cases and one sub-system (see figure X). I will now describe the writing system and briefly ellaborate on how the contained use cases support the scenario of writing a thesis. When possible, I will describe the use cases with a few phrases and only mention pre- and post-conditions when they are not apparent.

Use Case: Writing

To get started with writing, the student simply drags chapters from the SciFlow client's style drawer into the document, names them and re-arranges them. He can navigate through the document using a table of contents at the left hand side and drag and drop figures and references from both, the SciFlow bibliography manager and the asset manager.

Since the SciFlow writing client automatically saves the document to the harddrive and the server, the user can publish the document at any time.

SECTION 4.2

Creating the first draft document

To get started with writing, the student simply drags chapters from the SciFlow client's style drawer into the document, names them and re-arranges them. If the student were to publish the document at this stage - without applying any settings - he would receive a PDF that contains the title of the thesis, a table of

IMPLEMENTATION The SciFlow desktop client has been implemented in Adobe AIR to make it accessible on multiple platforms. With this requirement in mind, an implementation in Java (or similar) could have accomplished similar results.

Figure 1 presents a screenshot of the client. The most important user interface components are (1) the main text area, (2) the table of contents, (3) the style drawer and (4) a dedicated area for comments and figures to be displayed alongside the document. Furthermore, the (5) revision selector allows for quick access to previous versions of the document.

The user interface makes use of drag & drop techniques in multiple ways in order to make the interface more intuitive. Any of these actions is transparently relayed to the user; *i.e.*, these changes instantly appear in the user interface. If the user drags a reference from the asset manager or bibliography manager (not depicted) into the text (1), this will include a citation or figure at this position. A citation will appear directly within the text, while a figure will be displayed alongside the document (4). When the user drags figures from this area, a reference to the figure will be inserted into the text.

A user may re-organize the document by dragging sections within the table of contents (1) or make a reference to a section of the document by dragging a section from the table of contents into the text (1).

While the user interface has been kept simple by limiting the set of actions to a necessary minimum and designing towards intuitive use, a major requirement to scientific documents and documents in general is that they are protected from data loss or unintentional deletions.

3.2.3. Safety and Recoverability

In our survey, 82.2% of users performed backups rather often or very often (see Section 2.3). SciFlow provides two ways to support the user with document backups. For one thing, it incorporates revision management by taking snapshots of the document every two minutes. For another, these snapshots are pushed to the server, whenever a change is made and the user is connected to the internet. To keep the used bandwidth and disk space to a minimum the snapshots and their backups are made incrementally, meaning that only those parts of the document that have changed, are changed on disk. For a detailed description of the snapshot features please see [24].

However, since all SciFlow documents are also saved within the user's document directory, he may make manual backups at all times. Documents are saved in the DocBook XML format which may be read by a number of clients. In case of a hard drive failure the SciFlow client automatically retrieves backed up documents from the server. In the case of a software failure, the SciFlow client retrieves the last made snapshot. It may check the integrity of a document by comparing the files that are contained within a snapshot with the structure file it saves alongside the parts of the document. Should a file be missing from the snapshot, SciFlow may retrieve a version from an older snapshot and thus repair the file.

IMPLEMENTATION To perform backups, any web service that can store and retrieve files may be considered. In our implementation we have chosen Amazon S3 for several reasons. Amazon S3 files are mirrored in several data centers in multiple geographic locations. Furthermore, load tests conducted by [25] show that even when a single file is accessed excessively, it is still well accessible.

In order to maintain revisions of a document, SciFlow uses a copy-on-write snapshot implementation [27]. Copy-on-write, in contrast to redirect-on-write, describes a technique, where changes on data are always written on the original data. To keep a copy of earlier revisions, the original data is copied to a new location before the changes are saved.

3.2.4. Ease of Revision

In the previous section, we have introduced the way that SciFlow saves revisions of a document. This section will describe how a user may use these revisions to review and make changes to his document.

Each element of a SciFlow document (e.g., a paragraph, a figure, *etc.*) is assigned with a 128-bit unique identifier (UID). Such am element may exist in several forms. In the SciFlow client it is represented in the form of TextFlow, a markup language similar to XHTML. When a document is saved to disk, this TextFlow markup is transformed to DocBook XML with the addition of the UID attribute for each element. Once a document is exported, the SciFlow document publishing service transforms the DocBook XML first to LaTeX and then into a PDF using *dblatex*. The important thing to note is that the unique identifier is carried through all of theses forms. In the LaTeX source file a abbreviated form of the identifier is assigned to each document element in form of a label. We call this a *quick access code*.

If a user requires a print-out for review on paper, these labels may be printed alongside the document. We will assume that the user continues working on the document while a print-out is in review. Thus the document contents and structure may have changed, once the author wants to merge the manual comments made on the print-out. Since SciFlow remembers the unique identifier, it can find the position of a text element in the document, even when it has been moved within the document.

Alongside with the review on paper, SciFlow provides a review website which displays the document and allows for a reviewer to comment on text segments like paragraphs, sections or figures. These comments are automatically synchronized with the writing client and appear alongside the document.

IMPLEMENTATION The SciFlow review website is implemented as a Ruby on Rails application that may be started an arbitrary number of times and can serve any document that has been saved to the SciFlow storage service. To access it, a RSA encrypted key is necessary which is unique to the user who is given access.

In this section we have introduced the SciFlow concept. It is *scalable* because of its loose coupling and distributed architecture, features a *simple and transparent* writing client, so authors may focus on writing. Furthermore it is *safe* and provides *recoverability* since it constantly makes backups to the SciFlow storage service and it provides *easy revisions* with specially annotated PDFs and a review website. This concept has been implemented and also been used to write the thesis [24], providing a proof of concept. We will now elaborate on how the concept may be used by providing use case scenarios.

4. Application Scenarios for SciFlow

In this section we introduce two application scenarios for SciFlow. The first one is possible with the current state of the SciFlow writing tool, the second scenario demands for further enhancements of the SciFlow implementation. For illustration purposes, we will follow the example of Alice and Bob, two computer science students, Professor Davis and his research assistant Carol.

4.1. Writing a Thesis

The first document that was written with the implementation of the SciFlow concept was Frederik Eichler's master thesis [24]. To give an idea what the writing and review process for such an example looks like, we will now describe a simplified process from writing a first draft to the export of the final document.

We will follow the example of the grad student Alice. Assisting her are her thesis advisor Carol and her fellow student Bob. Upon completion, Professor Davis will read and mark her thesis.

Before Alice can start writing her document, Carol as one of professor Davis' research assistants **sets up the SciFlow environment**. To do that she selects the LaTeX style file they have already been using. This style file sets up the document margins as well as the citation style and the layout of the bibliography and will later determine the layout of the final thesis. Now she sends the link to the SciFlow writing client to her student Alice along with a username and an automatically assigned password. Alice opens the link and is guided through the installation process. Through this website Adobe AIR and the SciFlow client are installed within a minute. Once she has done so, she may log in with her credentials and start writing. Alice begins with typing some text and uses the styles drawer of the SciFlow client (see Figure 1) to **create her chapter structure and a first draft**. Once she has done so, Alice decides to consult Carol to be sure that the structure is set up right. She clicks *sharing* and enters Carol's email address. Carol now receives an email with the link to a **review website**. Since Carol is currently traveling back from a conference, she opens the link on her mobile phone and begins reading. When she sees something noteworthy, she uses the review website's comment function.

Alice has added the RSS feed for the comments to her E-Mail application, so when Carol starts reviewing the document, she gets notified and launches the SciFlow writing client. The client now starts synchronizing and **displays comments next to the paragraphs** they were meant for. Once she has worked through all comments, she marks them as *read*.

Bob prefers to **review** Alice's work **on paper**, so Alice clicks export and selects *for review*. She receives a link to a PDF which she prints out for Bob. Once Bob has annotated the document, Alice may use the *quick access codes* that are displayed alongside each paragraph to quickly find the text in the SciFlow client, even though she has modified the document and has moved sections around (see Section 3.2.4.)

While Alice is writing, her **hard drive crashes**, rendering the device useless. After a brief moment of panic, she remembers that the SciFlow client had notified her of a backup just a minute before the crash. She borrows her fellow student Bob's laptop and sets up the SciFlow writing software by again, following Carol's email. Only this time, when she logs in, the client begins to download the document she had been working on and all figures and references from the SciFlow asset and bibliography manager, she had saved already.

So far, Professor Davis has only seen a preliminary outline of the thesis. At the end of writing he now has to **review and grade** Alice's work, which adds to his workload of ten theses which he has had for review already. Before using SciFlow, Professor Davis often found himself in situations where he had time for review, but did not have the documents at hand. Now, he takes his e-reader (e.g., Amazon Kindle) with him, when he leaves the office. When he finds some time before a meeting or while in transit to work, he uses the commenting function of the e-reader to make notes on documents he has to review. Once back at the office, the device synchronizes his notes with the SciFlow software client.

4.2. Scientific Workshops

Besides the application of SciFlow in the context of the (mostly independent) writing of a thesis it could be applied to collaboration on writing, reviewing and rating of scientific papers. Therefore SciFlow would need to support basic means of collaborative writing. While synchronous editing may be helpful in some situations much of the cooperative scientific writing is asynchronous. In general, there are different co-authors responsible for different parts of a document, as stated in Section 2.1. The most basic collaboration feature for SciFlow would be to lock certain sections to be read-only by the co-authors. After they have completed their section they would release the lock and others would regain write access.

In this scenario, Carol and Professor Davis are to write a scientific paper about a new algorithm they developed at their university. The conference uses SciFlow as word processor and review tool. The organizers have already selected a convenient template file for the text and references and have pre-bundled them into a SciFlow version ready for download for all authors. Carol and Professor Davis would now need to create an account at the conference's conference system, download the SciFlow client and log onto their account. After Carol and Professor Davis have brainstormed about ideas how to communicate their new findings, Carol sets up the basic outline of the paper in SciFlow and assigns different parts to herself and her professor. Both of them write their respective parts and use the built-in review mechanism to enhance the quality of their respective texts and to stick to the golden thread of the paper. During the writing process both of them can rely on the safety of their work, because the conference system backs up all their data every time they make chances to the document. The customized version of SciFlow shows the remaining time until the submission deadline and once they finished their work on the paper, Carol submits the paper from within SciFlow.

The organizers view in the conference system allows to track the number of started papers, the number of pages and recent updates from each of them. The organizers can set the submission deadline from within the conference system, thus blocking the SciFlow client from submitting papers after the deadline. Furthermore, the conference system automatically extracts keywords from each document and generates word clouds from the papers to facilitate the assignment of appropriate reviewers for each submitted paper. The reviewers of Carol's and Professor Davis' paper received an automatically sent notification about the upcoming review task and a link to the SciFlow download. The reviewers can use the SciFlow client or the review website for accomplishing their task and their rating of the paper. After the acceptance of Carol's and Professor Davis' paper they can see the annotations for their paper at the corresponding place in the text and can improve their text until the final submission of the camera-ready version of the paper.

5. Conclusions and Outlook

5.1. Discussion of the Presented Approach

Within the previous chapters we described the SciFlow concept. It simplifies scientific writing and review by taking advantage of templates and cloud computing to make a slim writing client possible. Furthermore, we have described two scenarios that explain how the SciFlow concept can be used for a thesis and a scientific workshop scenario. However, the number applications for this concept is great. [24] describes a number of these applications including collaboration and peer review, keyword extraction, interactive PDFs, text-to-speech and video review, just to name a few. The choice of DocBook XML as an underlying format and the scalable asynchronous communication between multiple services, encourages the integration of these and more features. At the same time the use of web services in the SciFlow architecture allows for thin writing clients that may even be deployed to mobile devices. Moreover, DocBook XML and its available validation schemata (e.g., DTD, XML Schema) allow for new services to be connected without introducing the possibility of rendering existing data inconsistent. While the existing implementation was built with this validation in mind, this is one of the areas of improvement of the software. SciFlow will be extended to allow the co-creation of various other scientific documents such as research articles, book reviews or case-reports and therefore we will need to reconsider the choice of DocBook XML. There are other XML DTDs on the market that are more specifically tailored to the creation of scientific writings as mentioned above (e.g., the Journal DTD [28]) which in turn do not support the creation of theses.

With the use of external services for data storage come concerns about data ownership as well as legal issues. To address the former, the SciFlow software architecture was designed to function with a great number of services or even custom servers in a data center of choice. Each service comes with a pre-defined interface and its server software is executable on a number of Linux/Unix distributions (e.g., Mac OS X, Debian Linux, Gentoo Linux). Those services may be integrated into the existing architecture as introduced in Section (see Scalability).

The choice of Amazon Web Services presents one possible scenario for an implementation; one where Amazon's service level agreements [29] are deemed sufficient and there are no legal reasons for not choosing Amazon or a similar cloud service. Legal requirements may be influenced by University and company policies or even the personal opinions of an executive (e.g., CIO, head of department, *etc.*). Another concern may be that, to a certain extent, one will be bound to obey by further pricing changes by the cloud service provider. The two choices at hand here are to either invest into an own IT infrastructure to control the architecture into the last detail or make use of service providers and thus accept the SLAs provide by them. Technologically this makes no difference to the SciFlow concept; legally and financially the implications may vary.

5.2. Future Research and Development Opportunities

For now, we will describe two ideas that may significantly improve the benefit SciFlow offers to its users. Section 4.1 already introduced the possibility to use an e-reader to review documents. During development we have experimented with the export of SciFlow documents to the *MOBI* format [30].

This format in particular is used by devices like the Amazon Kindle [31]. When a user annotates such a document on his device, these comments are saved in file that may be accessed via the usb interface of the Kindle. Thus it would be possible to recover comments that were made without the user's manual intervention.

Since Adobe Acrobat 9, all Acrobat Reader products are equipped with a Flash Runtime environment [32]. In the case of this very paper interactive content would have allowed us to include multiple screenshots of the SciFlow client into an interactive version of the document. In this case, SciFlow users would be able to define a gallery of images and choose a representation of the gallery that appears within print-outs. The non-interactive print-out might have a url added to the description of this representation so that all readers can access the content. This would be similar for video content.

In summary, we have presented you with a scalable and easy to use concept, a concept that is safe. We have presented you with a concept that supports online review and especially annotated PDFs. All that comes in a 2 MB large desktop client, that can be installed within minutes and needs no setup by the writer. The SciFlow concept remains to be tested with a larger user-base, e.g., in a seminar or workshop to determine areas of improvement as well as an extension towards the requirements of collaborative writing.

References

1. Baecker, R.M.; Nastos, D.; Posner, I.R.; Mawby, K.L. The User-centred Iterative Design of Collaborative Writing Software. In Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, Amsterdam, The Netherlands, 24–29 April 1993; ACM: New York, NY, USA, 1993; pp. 399–405.
2. Noel, S.; Robert, J.M. Empirical Study on Collaborative Writing: What Do Co-authors Do, Use, and Like? *Comput. Support. Cooperat. Work* **2006**, *13*, 63–89.
3. Grover, C. *Word 2007: The Missing Manual*; O'Reilly Media: Sebastopol, CA, USA, 2006.
4. Liu, J.; Sadler, R.W. The effect and affect of peer review in electronic versus traditional modes on L2 writing. *J. Engl. Acad. Purposes* **2003**, *2*, 193–227.
5. Academic Software Zurich. Citavi Reference Management Software. Available online: http://www.citavi.com/ (accessed on 2 December 2010).
6. Center for History and New Media, George Mason University. Zotero. Available online: http://www.zotero.org/ (accessed on 2 December 2010).
7. Google. Google Scholar. Available online: http://scholar.google.com (accessed on 2 December 2010).
8. Ritterbush, J. Supporting Library Research with LibX and Zotero. *J. Web Librarianship* **2007**, *1*, 111–122.
9. Oracle. Open Office. Available online: http://www.openoffice.org/ (accessed on 2 December 2010).
10. Rossi, B.; Russo, B.; Zuliani, P.; Succi, G. On the Transition to an Open Source Solution for Desktop Office Automation. *Lect. Note. Comp. Sci.* **2005**, *316*, 277–285.
11. Lamport, L. *LaTeX: A Document Preparation System*; Addison-Wesley Professional: Reading, MA, USA.
12. van der Hoeven, J. GNU TeXmacs. *SIGSAM Bull.* **2004**, *38*, 24–25.

13. UIUC REU Number Theory Program. Introduction to LaTeX. Available online: http://www.math.uiuc.edu/ hildebr/tex/course/intro1.html (accessed on 2 December 2010).
14. Patashnik, O. Designing BibTeX Styles. Available online: http://www.eeng.dcu.ie/local-docs/btxdocs/btxhak/btxhak/btxhak.html (accessed on 2 December 2010), 1988.
15. Google. Create and share your work online with Google Docs. Available online: http://docs.google.com (accessed on 2 December 2010).
16. Dekeyser, S.; Watson, R. Extending Google Docs to Collaborate on Research Papers. Available online: http://www.sci.usq.edu.au/staff/dekeyser/googledocs.pdf (accessed on 2 December 2010).
17. Sterken, C. Advice on Writing a Scientific Paper. In *Astrophysics of Variable Stars*; Astronomical Society of the Pacific: San Francisco, CA, USA, 2006.
18. Adobe. Adobe Buzzword. Available online: http://www.adobe.com/acom/buzzword (accessed on 2 December 2010).
19. Gopen, G.; Swan, J. The Science of Scientific Writing. Available online: https://www.american scientist.org/issues/id.877,y.0,no.,content.true,page.1,css.print/issue.aspx (accessed 2 December 2010).
20. Skern, T. *Writing Scientific English: A Workbook*; UTB: Stuttgart, Germany, 2009.
21. Aragon, T.J. Practical LATEX for Public Health and Medicine. Available online: http://www.medpi.net/aragon/ (accessed on 2 December 2010).
22. Feruglio, G.V. Do Journals Honor LATEX Submissions? Available online: http://www.tug.org/TUGboat/Articles/tb17-2/tb51vali.pdf (accessed on 2 December 2010).
23. Hwang, A.D. Writing in the Age of LATEX. *Notices AMS* **1995**, *42*, 878–882. Available online: http://www-ljk.imag.fr/membres/Bernard.Ycart/writing/hwang.pdf (accessed on 2 December 2010).
24. Eichler, F. Simplifying The Scientific Writing And Review Process. Master's Thesis, University of Paderborn, Paderborn, Germany, 2010.
25. Onibokun, A.; Palankar, M. Amazon S3: Black-Box Performance Evaluation. Available online: http://www.csee.usf.edu/ aoniboku/Docs/AmazonS3_ProjectReport(2).pdf (accessed on 2 December 2010).
26. Amazon. Auto-scaling Amazon EC2 with Amazon SQS. Available online: http://developer.amazon webservices.com/connect/entry.jspa?externalID=1464&categoryID=177 (accessed on 2 December 2010).
27. Brinkmann, A.; Effert, S. Snapshots and Continuous Data Replication in Cluster Storage Environments. In Proceedings of Fourth International Workshop on Storage Network Architecture and Parallel I/Os (SNAPI 2007), San Diego, CA, USA, 24 September 2007.
28. Journal Publishing Tag Set Tag Library version 3.0. Available online: http://dtd.nlm.nih.gov/publishing/tag-library/ (accessed on 2 December 2010).
29. Amazon. Amazon S3 Service Level Agreement. Available online: http://aws.amazon.com/s3-sla/ (accessed on 2 December 2010).
30. MobileRead. MOBI. Available online: http://wiki.mobileread.com/wiki/MOBI (accessed on 2 December 2010).

31. Amazon. Amazon Kindle. Available online: http://www.amazon.com/dp/B0015T963C (accessed on 2 December 2010).
32. Adobe. Adobe Acrobat Family: Product Comparison. Available online: http://www.adobe.com/products/acrobat/matrix.html (accessed on 2 December 2010).