IDDS: An Interactive Decentralized Documentation System

Christoph Meinel, Harald Sack, Volker Schillings
FB IV - Informatik, Universität Trier
D-54286 Trier, Germany
{sack, meinel, schillin}@ti.uni-trier.de

Abstract

The paper describes the design and the application of an Interactive Decentralized Documentation System (IDDS). IDDS is a web-based, interactive documentation system that is especially designed for the support of working groups distributed over many places. It enables the creation of "just-in-time"-documentation and provides authoring tools for multimedia documents supporting multiple authors per document including a versioning system. IDDS provides an HTML-frontend for being accessible anywhere with a standard web-browser for creating, reading, or commenting on the provided documentation. The workflow of IDDS includes an interactive and guided reviewing process of the documentation including security mechanisms that are designed to maintain a high quality of the created documentation. Furthermore, first experiences of the application of IDDS in a system and network-administration environment are given.

Keywords: documentation system, distributed authoring, multimedia documentation, distributed documentation processing

1. Introduction

In all areas of technical research and development, as well as for any other subject that is related to the development of complex tasks with the requirement of being reproducible and being maintainable, the acquisition of an appropriate documentation is absolutely mandatory. Esp. for complex tasks as software development or system administration nobody doubts the benefits of a well structured, diligently written, and unrestricted accessible documentation. Without appropriate documentation, maintenance tasks related to these subjects often develop to entire reengineering projects and soon become much too expensive.

Unfortunately, although everybody knows about these fundamental requirements for documentation, it is neglected rather often, because in most cases the problem solution itself requires the entire attention, while documentation is being postponed to a later point in time, when the critical development parts of the project are already finished. But often, development continues until the very last deadline is reached and there will be no time left for an appropriate documentation. People, who are assigned to solve problems, should be enabled to create also documentation on the flight, without much overhead, just in time, while acquiring the solution of the designated problem, at any time and any place. No sophisticated management tool dependent on specific hardware or operating system should be the basis for the documentation process, moreover documentation should be created by applying standard tools available on every possible computing platform connected to the internet.

On the other hand, the quality of the gathered documentation is often a crucial point. If the documentation is not subject to any reviewing process, it is hard to avoid mistakes and thus, to maintain the required quality of the documentation. Also, for large projects, sophisticated management of distributing the single documentation tasks over a group of several experts is rather important. For this reason, it is significant to maintain also proper and reliable versioning, because it should be always clear, who is responsible for which part of the documentation.

Communication on the global scale, enhanced by the tremendous growth and the popularity of the World Wide Web (WWW) are giving way for decentralized, web based approaches that enable every member of a working group to participate interactively in the development of an appropriate and certified documentation that will be immediately accessible world wide. Thus, the focus of this paper lies on the development and deployment of an Interactive Decentralized Documentation System (IDDS). IDDS is a web-based documentation system that is especially designed to meet these requirements.

Thus, IDDS gives way to the development of documentation on the flight with an automated and decentralized reviewing process. IDDS enables faster creation of certified and well structured documentation, fulfilling the requirements of today's early time-to-market schedules. Similar documentation systems do already exist on the WWW, as. e.g. SourceForge [9], a large open source documentation

To be published in Proc. of ACM SIGDOC 2001, Oct. 2001, Santa Fe, New Mexico, USA

site. But, IDDS enables an even better integration of multimedia objects into the documents and, for our subject more important, provides better administration tools for security management.

To show the capabilities of IDDS we present an example application, where the system is used for the documentation of system administration tasks. The department of Theoretical Computer Science and New Applications, together with the affiliated Center of Electronic Publishing [2] at the University of Trier provided an excellent testbed for this task.

The paper is structured in the following way: In Section 2, we try to comprise all key properties of an efficient and successful documentation system. Section 3 introduces IDDS with the main focus on its implementation. Section 4 gives an example of the deployment of IDDS in a system administration environment and gives proof of its success. Section 5 finally concludes the paper by summarizing the key results and by giving an outlook on future research and development of IDDS.

2. The "Ideal" Documentation System

Before the development of a new documentation system starts, first, it is important to comprise all critical properties that need to be achieved. For this purpose, we take a short look on the documentation process of an arbitrary complex system development project. To guarantee an appropriate, qualified, and useful documentation, we have to consider the following requirements:

Start before it is too late - the time requirement: documentation is to be developed during the problem solution process. Of course, it may be necessary to adapt the documentation, whenever an error will be trapped within the system under development. But, if documentation is only written after the problem solution process has finished, important issues might get lost easily because they are already forgotten by the system developers, who are writing the documentation. Another important point is that tight time schedules often don't give enough room for a proper documentation, because system development is carried on until the very last minute and documentation is only supplemented in a way that might be called *quick-and dirty*. Thus, the best way to develop appropriate and qualified documentation is writing documentation *just-in-time*, or *on-the-flight*.

Don't give away the documentation task - documentation is groupware by nature: in the development of complex systems often many people are involved in implementation and problem solution. Often these people share not the same working place and might be distributed over large distances. If the documentation is postponed and given to a separate centralized documentation group of technical writers, the result simply cannot be of the same quality as if it had been written, reviewed, and revised by the system developers themselves during the development process. People, who are solving problems know best about their own problem solutions. Giving documentation away to professional technical writers means to add another layer between the expert and the consumer of the documentation and gives way to additional errors to be included. Therefore, documentation should be written, reviewed, and revised by the people, who are responsible for the development of the system to be documented.
Of course it is also necessary to take care for the reader of the documentation. What is his/her background and how much technical details are necessary or required. But this question concerns readability of documentation and will not be covered in this paper. In the following we assume that experts are writing documentation for experts. If the documentation is written for a less qualified community, then the authors must consider this carefully, or they include professional technical writers, people with the required knowledge about writing documentation for the particular target group.

Panta rhei (everything flows) - documentation changes during system development: If we stick to the premise to write documentation on-the-flight, it might be necessary to refine, to change, or to eliminate parts of the already existing documentation. For this reason and also by considering the other premise that many people are involved in developing documentation, a strict rights management has to be kept. In addition to that, also a transaction management has to be included, that a document can only be subject of one single change at a time. Additionally, for keeping a record about the changes in documentation, the evolution of the documentation project has to be recorded in some history list. There, it should be possible, to find out, what has been changed for each single version of a document, who is responsible for that change, and of course one should also be able to switch back to a previous version of the document.

A picture says more than a thousand words - documentation consists out of multimedia elements: The term documentation is not only restricted to written text files. As the title of this paragraph says, often it is much easier to explain a specific context by providing a single picture, or even a collection of annotated pictures. Not only that, sometimes, it is even of advantage to listen to the expert himself by including some audio recording to the documentation. And if even more clarification is required, even video clips or animation should be included into the documentation file. In addition to that, also interactive elements as e.g., Java-applets might get included. Therefore, a good documentation framework should be able to handle all kinds of multimedia elements.

Don't try to keep everything always at one place - parts of the documentation might be largely distributed:

Well, this point might raise kind of a controversy. On the one hand, if you try to keep the entire documentation collection at a central server, control and maintenance tasks are much more efficient to manage. On the other hand, changes and updates of specific elements out of the documentation always require the central server to be accessed. If parts of the documentation are simply copied from a remote location, somebody has to keep track of these changes and has to keep the central documentation to the latest release. Also a drop out of the central server has fatal consequences for a centralized documentation system. On the other hand, the world wide web offers the possibility to include information via hyperlinks into documentation that is locally not available. Also, in case of failure or dropout, only parts of the documentation might get not accessible. By engaging mirror servers, even this risk might get minimized by providing a suitable number of mirrored information. For these reasons, in our opinion, for large documentation projects, including lots of multimedia elements, distribution of information comes quite natural.

The customer might be located anywhere in the world - access of documentation is location independent: Good documentation should be available everywhere and at anytime. No matter, where you are located, you should be able to access documentation as long as you are able to access the internet. This can range from a standard PC and a wired network connection at work to a PDA connected to a wireless network at any place, where a connection is possible. To keep the access as simple as possible, a standard web-browser as front-end should be used instead of a hardware dependent device relying on a specific operating system. Thus, access to the documentation should be possible from everywhere via the WWW.

Good quality is hard to maintain - the documentation process has to be supervised: Whenever a group of people are involved in creating many documents, somebody must take care that a certain level of quality is properly maintained. This can be achieved by pointing out a supervisor from of the group of authors, who from now on will be responsible for what is being published and what not. Before anything gets published, he has to decide, if the documentation meets the mandatory quality standards, or if any revision is necessary. High quality can only be maintained, if the reviewing process is subject of high attention. But, there exists also a tradeoff between quality and time requirements, as there is for all kind of scientific publications [1]. Often, there is only little time and documentation has to be published rather quickly to be of any use. To accelerate the process of publication one can also grant an automatic certificate of approval, if several authors agree upon a document to be published. As an additional task, the supervisor has to distinguish, who is capable of accessing what kind of information. Some of the documentation might be recommended for the overall public, but other documentation should only be available for a restricted group of users. Thus, a sophisticated access management for maintaining high quality standards is mandatory.

To hear different opinions is often worth while - documentation should be annotated: When including several people, who are responsible for the development of documentation, sometimes there might not be consensus about certain subjects. Thus, if another expert is reading the document of an author, he feels the urge to add some comments, also valuable for the public. On the other hand, also consensus should be included, because it makes the documentation more trust worthy. Thus, our documentation system provides the possibility of adding comments on documents.

Cave canem (absolute free translation: Beware of any unauthorized access) - Security issues: Of course not all documentation is meant for public use. Not only unauthorized read access must be prevented, because some documentation is restricted to a selected group of users only. On the other hand, unauthorized write access must also be prevented to maintain the quality, the reliability, and the quality of the documentation. Whenever documentation is used, also the question for liability turns up. The liability is always on the side of the documentation provider. Therefore, the documentation provider has to design specific security guidelines and has to watch that they are strictly kept.

It must be useful - data retrieval facilities for documentation: Developing documentation for electronic media often requires much more effort than just preparing printed material. The inclusion of animations or video/audio clips requires also the investment of additional effort for their creation. But, if the documentation is available in an encoded format within a computer, the next step is the processing of the available data with the help of database management systems. There, not only a standard full-text search is possible, but also sophisticated retrieval techniques can be used to facilitate better access to the required information with little additional work. Thus, whenever developing large documentation systems, one should make use of available data retrieval facilities to gain additional added value.

3. IDDS - Idea and the Implementation

With the Interactive Decentralized Documentation System, our working group has developed a powerful documentation system with the goal to achieve as many of the previously cited properties as close as possible with the restriction of investing only as little manpower as necessary.

In the design of IDDS we defined three different role models related to different processing tasks for the process of creating a new documentation. First let us start with the tasks of processing the documentation:

Accessing documentation: every user, who has the required access rights for the requested documentation is able to read the documentation with a standard browser in every computing environment. Publicly available documentation needs not to be guarded according to read-only access. Access rights will be granted by the documentation's administrator and can be demanded by every user.

Authoring documentation: all designated authors have the right to create new documents within a documentation project. Authoring rights will be granted by the documentation's administrator and can be demanded by every user. Every author has the right to revise, to extend, or to delete his documents. The revision of a document that is authored by another author within the same documentation project can only be demanded by the author. Write-access will then be granted by the documentation's administrator or by the document's author. The author cannot further grant his write-access to other authors.

Approving documentation: before any document gets published, it has to be approved. Approval can be granted either by the documentation's administrator, or by another author, who was requested for approval of this document.

Publishing documentation: all approved documents of a documentation project can be published by the documentation project's administrator. He is the final instance, who decides what is going to be published and who is able to access the published documents. For accelerating the publishing process, also the consensus of several authors - in terms of several independent approvals on a document - can be made responsible for a publication.

Commenting documentation: everybody with read-access is able to comment on a document. This refers to already published documents as well as for unpublished material that is still within the working cycle of the documentation project. The author of a document will be notified of any comments about his work and is demanded to process the comment accordingly. Processing a comment can lead to a revision of the document, to another comment of the author, or simply to its deletion. Comments that remain within the document will also be subject of publishing.

This processing cycle of a documentation project includes three different role models, which are

- the **administrator**: his tasks start with the creation of a new documentation project. The next step is the assignment of people of a special working group (experts) to the development of the new documentation project. He grants write-access to the designated experts of the documentation project. He is the final instance for publishing documentation, giving a certificate to already finished and approved documents. He grants read-access to published material according to the security guidelines of his principal.

- the **experts:** experts are members of an assigned working group, who's tasks include all processing steps in the creation of documents for the documentation project. This starts with the creation of new documents. Then, there follows a possible cycle of revisions and comments that have to be included into the document. If requested, experts can also contribute to documents of other experts. Besides contributing additional content, experts can be requested to give approval to finished documents. Experts can demand to contribute to other documents or can give comments on other documents at any time.

- the **consumers:** everybody, who has read-access to documents of the documentation project can be regarded as a consumer of that documentation. If no read-access is granted, the consumer can request permission from the documentation project's administrator. Every consumer is free to comment on all available documentation that he is able to access. If a consumer feels that he has more to contribute than just a comment, he can request to be promoted to the group of experts by the documentation project's administrator.

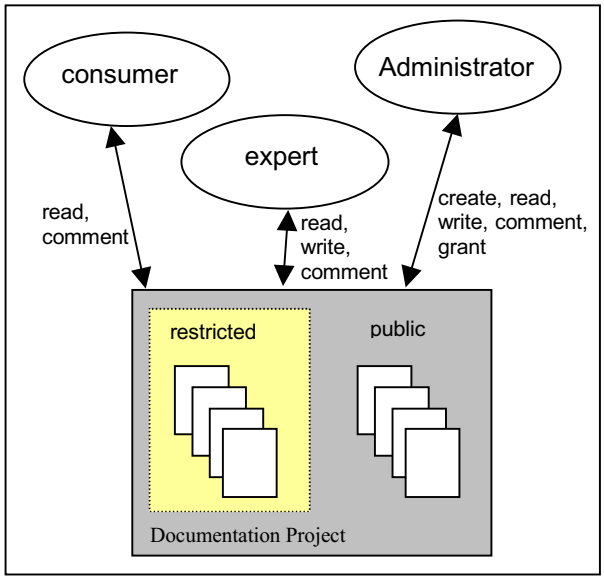

Fig.1.: IDDS Role Model

See Fig. 1 for an outline of the IDDS workflow and the role model.

Documents within a documentation project can be in one of the following states:

- **Working draft**: all documents that don't have any approval, neither by another expert nor by the administrator, are in the state of a working draft.

- **Approved document**: all working drafts that got approval by another expert or by the administrator are approved documents.
- **Certified document**: approved documents that are granted to be published by the administrator or by approval of several experts are certified documents.

To provide reliable versioning, a history record is kept for every document. All changes are logged and if it is required to switch back to an old version, the author or the administrator are able to do this with the history record.

IDDS provides a HTML-based front-end for all kinds of the above mentioned user. All displayed pages consists out of dynamic components that are created using **perl v5.005** for i586-linux [4], **javascript** [5], and **java v1.1.8** [6]. For providing access with a standard web-browser, java has to be enabled in the browser's setup. Thus, enabling a working system independent of hardware and operating system requirements. All documents and user related data are stored in a relational database management system

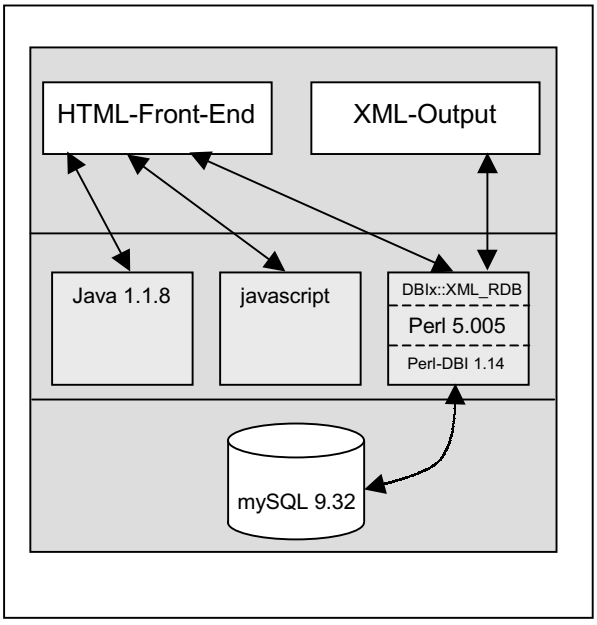

Fig. 2.: Multi-tier Architecture of IDDS

using the database's own security provisions. We chose the publicly available **mySQL** v9.32 system [7] that is available for most operating systems and hardware platforms. As an interface for the access to the database with perl serves **perl-DBI** v1.14 [8]. For generating XML output [3] that might be required for any kind of postprocessing, the perl extension **DBIx::XML_RDB** is used, which directly supports the creation of XML from existing DBI datasources.

Thus, a 3-tier architecture of IDDS is maintained, consisting of a *user/application-layer* that consists out of the HTML-Front-End, serving as the user interface and an XML-Output, which can be used for postprocessing options.

The second layer, the *application-layer* consists out of the single application modules, as shown in Fig. 2. The third layer, the *data-layer* contains the mySQL database that is accessed by the perl-DBI interface in the application-layer.

The administration of users - here the administrator, the experts, and the consumers - is accessed by standard HTML-form-handles and communicates directly with the database. Each authorized user - here the administrator, experts, and consumers, who are granted for restricted access areas - has to identify himself to the system with a username and a password to prevent unauthorized access.
Experts are creating a new document with a HTML-form-handle and can edit the document with an WYSIWYG-editor, implemented in Java. The editor allows the inclusion of hyperlinks, pictures, audio-, and movie-clips, as well as the inclusion of java-applets into a text document. Pictures are included as a thumbnail of the original image that is connected to the original image via a hyperlink. Audio- and movie-clips are represented by an appropriate icon, connected to the original media-file via hyperlinks. All resources that are part of a document will be stored in the database and a history record will be kept, each time tha data is changed.

4. Working with IDDS

The implementation of IDDS was developed on a Linux system. Because only standard software tools are employed, it can be easily transferred to other operating systems.

As an example of the application of IDDS, we have chosen to document the system administration of our department. There, a lot of PCs running Linux and Windows, including older SUN Workstations running SunOS, and several peripherial devices have to be documented. Important issues as hardware configuration as well as software configuration are subject to documentation.

In the beginning, the administrator - here the system administrator of the department - creates a new documentation project and assigns the group of experts, including himself and some of our coworkers, by granting them read-write access to the new documentation project. See Fig. 3 for a list of available documentation topics and Fig. 4 for a list of documents, related to a specific documentation project. Additionally he prepares a list of single topics related to the documentation project. Here, for the system administration project, we have e.g. the installation of

a new Linux operating system version, or the installation of a new wireless LAN.

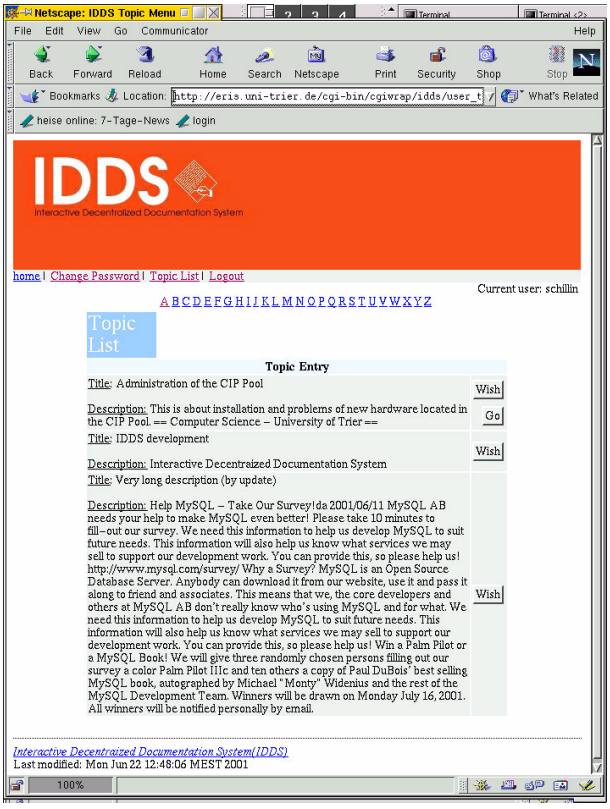

Fig.3.: Topic List of Documentation Projects

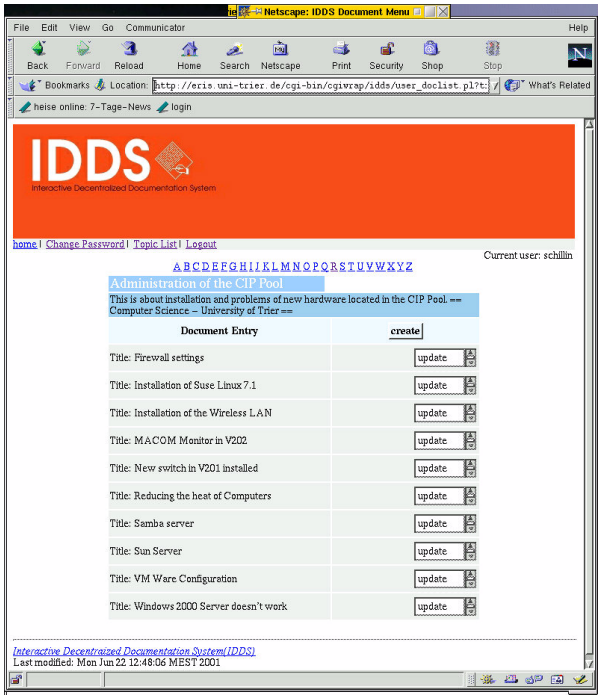

Fig.4.: Document List for a Specific Topic

Now, an authorized expert can login and select one of the topics within the documentation project that he is authorized for. See Fig. 5 for an overview of the access rights management.
Having selected a topic, the expert is able to create a new document, to edit a document, for which he has write access, to approve a document, if he has been asked for, or to comment on any document. He has the possibility to include hyperlinks or any multimedia object. See Fig. 6 for the editing of a document in IDDS.

If a document has got approval, the administrator can grant a certificate for publication. According to the previously created access-right list, from now on the document will be available on the web.

To view a document, a customer has to login via username and password, if required, and then is able to browse documents or to comment on documents (see Fig. 7).

Comments are added to documents in terms of comment threads. If a revision of a document is necessary, the revision number will also be displayed with the document.

IDDS is successfully deployed in our department and is also about to be used in large scale software projects, where it has to give further proof of its efficiency.

5. Conclusions and Future Work

We have presented the implementation of a documentation system that allows interactive development of any kind of technical documentation. Authors and users of the documentation participate both in the development process of documentation, which gives way for maintaining high quality standards. The implementation is provided with an HTML-interface that allows access of the documentation from everywhere with a standard web-browser available in every kind of computing environment.
But, as for any kind of documentation system, the quality of the supported documentation is highly dependent on the users' acceptance of the provided framework. Thus, we tried to make the development process as simple as possible and being already usable during problem solution.
Future work is concerned about the failover capacity of IDDS. Therefore, the data related to the documentation project should not be stored on a central server, but might get distributed over the web, which requires additional administrational effort for providing data consistency. A viable approach can be seen in peer-to-peer computing, where a central server only stores index tables associated with the actual location of the maintained data. The data itself is distributed over the connected peers and can be directly accessed from them. There, consistency must be managed starting from the central index server. To provide even more

security, this index server can further be mirrored at a distant location.

6. Acknowledgements

The authors like to thank Steve Poon for his effort in the implementation of IDDS in our computing environment and for his ideas and suggestions that have been integrated into the system.

7. References

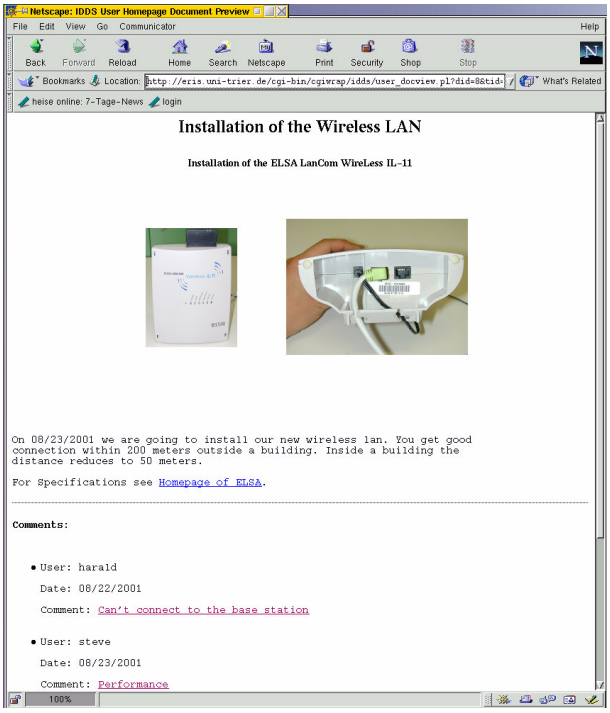

Fig.7.: Viewing a document

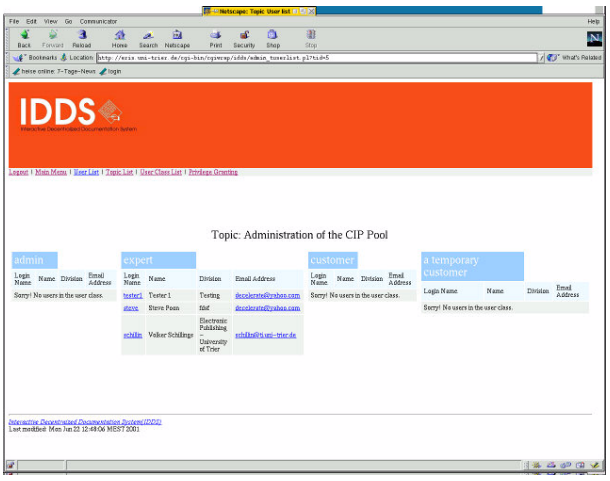

Fig.5.: User Access Rights Management

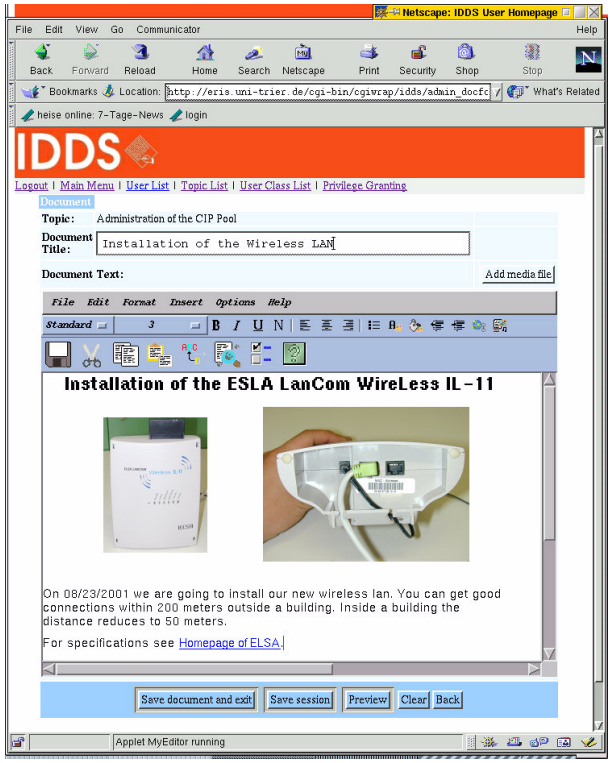

Fig.6.: Editing a Document

[1] J. Bern, Ch. Meinel, H. Sack: Electronic Colloquia: Idea and Practice, in Proc. of ACM SIGDOC 1998, Quebec City, Canada, 1998, pp. 113-119

[2] Center for Electronic Publishing (WEP) University of Trier [Online material], available http://www.wep.uni-trier.de/

[3] Bray, T. et al. (ed.): Extensible Markup Language (XML) 1.0, W3C [Online material], available http://www.w3c.org/

[4] See http://www.perl.com/ for online available documentation on perl

[5] See ftp://ftp.ecma.ch/ecma-st-Ecma-262.pdf for online available documentation on javascript

[6] See http://java.sun.com/products/jdk/jdk1.1/docs.html for online available documentation on java

[7] See http://www.mysql.com/ for online available documentation on mySQL

[8] See http://safari2/oreilly/com/table.asp?/bookname=perldbi&cnode=69 for online available documentation on perl-DBI

[9] See http://www.soujrceforge.net/ for online available documentation on SourceForge