

University of Applied Sciences of Fribourg
Diploma Project

Extending CDSware with social tools

Gregory FAVRE

Keywords CDSware, Digital library systems, Social software,
Collaborative tools

Dates September, 26th— December, 3rd, 2005

Supervisors Omar ABOU KHALED (EIA-FR - TIC)
Houda CHABBI DRISSI (EIA-FR - TIC)
Jean-Yves LE MEUR (CERN)
Tibor ŠIMKO (CERN)

A leader is someone who steps back from the entire system and tries to build a more collaborative, more innovative system that will work over the long term.

Robert Reich, United States Secretary of Labor
(1993–1997)

Preface

Abstract

English

This report describes my diploma project, ending my studies at University of Applied Sciences, Fribourg. This work has been realized at the European Organization of Nuclear Research (CERN), in Geneva. The aim of the project was to enhance CDSware, a digital library software developed by the CERN, with collaborative features.

Français

Ce rapport décrit mon travail de diplôme, réalisé au sein de l'Organisation Européenne pour la Recherche Nucléaire (CERN), à Genève, dans le cadre de mes études à l'École d'Ingénieurs et d'Architectes de Fribourg (EIA-Fr). Le but de ce travail est de développer et d'adjoindre des outils de collaboration à CDSware, logiciel de gestion de bibliothèque numérique, développé et maintenu par le CERN.

Deutsch

Dieser Bericht beschreibt meine Diplomarbeit der Hochschule für Technik und Architektur, Freiburg, die im Europäischen Zentrum für die Kernforschung (CERN) in Genf stattgefunden hat. Das Ziel dieser Arbeit besteht darin, Werkzeuge für Zusammenarbeit in CDSware, eine digitale Bibliotheksoftware, zu entwickeln.

Italiano

Questa tesi si riferisce al mio progetto finale di Laurea, realizzato presso il Centro Europeo per la Ricerca Nucleare (CERN), nell'ambito dei miei studi presso la Scuola d'Ingegneria e Architettura dell'Università di Friburgo. Lo scopo del mio lavoro è di sviluppare dei sistemi collaborativi all'interno di CDSware, un'applicazione per la gestione di librerie digitali, gestita e mantenuta presso il CERN.

Conventions

Some signaletic symbols are used in this report



This icon points out coding. The code is formatted according to the language it is written in.



This icon points out related information, generally quoting a source on the Internet.

Contacts

Supervisors at EIA-FR

Omar Abou Khaled

École d'ingénieurs et d'architectes
de Fribourg
Bd de Pérolles 80 – CP 32
CH-1705 Fribourg
Phone: +41 26 429 65 89
Fax: + 41 26 429 66 00
Email: Omar.Aboukhaled@hefr.ch

Houda Chabbi Drissi

École d'ingénieurs et d'architectes
de Fribourg
Bd de Pérolles 80 – CP 32
CH-1705 Fribourg
Phone: +41 26 429 65 89
Fax: + 41 26 429 66 00
Email: Houda.Chabbi@hefr.ch

Supervisors at CERN

Jean-Yves Le Meur

CERN / Division ETT
CH-1211 Genève 23
Phone: +41 22 767 47 45
Fax: + 41 22 766 92 77
Email: Jean-Yves.Le.Meur@cern.ch

Tibor Šimko

CERN / Division ETT
CH-1211 Genève 23
Phone: +41 22 767 35 27
Fax: + 41 22 767 65 55
Email: Tibor.Simko@cern.ch

Student

Gregory Favre

rte de Châtel 19
CH-1272 Genolier
Phone: +41 76 452 99 87
Fax: + 41 22 366 24 01
Email: Gregory.Favre@gmail.com

Contents

Preface	iii
Contents	vii
1 Introduction	1
1.1 CERN	1
1.2 CDSware	1
1.2.1 Context	1
1.2.2 General overview	2
1.2.3 Information sharing	2
1.3 Social software	2
1.4 Work	3
1.4.1 Tasks	4
1.5 Report organization	4
2 Study of CDSware	7
2.1 Users and access	7
2.2 Features	7
2.2.1 Search	7
2.2.2 Document submission	9
2.2.3 Current personalization features	9
2.3 Technologies	9
2.3.1 Programming languages	10
2.3.2 Web server	11
2.3.3 Database	11
2.3.4 The underlying bibliographic format	11
2.3.5 OAi: Open Archive initiative	11
2.3.6 Discussion	12
2.4 Architecture	14

2.4.1	General organization	14
2.4.2	Programming methodology	15
2.4.3	Modules	15
2.4.4	Operation principles	18
3	WebMessage	21
3.1	Specifications	21
3.2	Analysis	21
3.2.1	User interface	22
3.2.2	Database	23
3.2.3	Architecture	26
3.3	Realization	27
3.3.1	Database	27
3.3.2	Scalability tests	28
3.3.3	User interface	30
3.4	Conclusion	32
4	WebBasket	33
4.1	Specifications	33
4.2	Analysis	33
4.2.1	User interface	33
4.2.2	Database	34
4.2.3	Architecture	40
4.3	Implementation	41
4.3.1	Database	41
4.3.2	User interface	41
4.4	Conclusion	43
5	General conclusions	45
5.1	Results	45
5.2	Improvements	46
5.3	Personal results	46
	Bibliography	49
	Glossary	51
A	Initial specifications of project	53

B Paulo Cabral's prototypes	59
B.1 Database	59
B.2 WebMessage	61
B.3 WebBasket	66
C Complete database diagrams	79
Entity-Relationship diagram	80
Relational diagram	81
D API	83
D.1 WebMessage	83
D.2 DateUtils	85
D.3 MailUtils	88
D.4 TextUtils	90

Chapter 1

Introduction

1.1 CERN

The European Organization for Nuclear Research (CERN) is the world's largest particle physics center[1]. Physicists at CERN explore what matter is made of and what forces hold it together. This organization provides them the necessary tools for their research. These are mainly accelerators able to bring particles to almost the speed of light, and detectors to make the particles visible.

Founded in 1954 by 12 countries, it has grown to the present 20 member states. Some 6500 visiting scientists, half of the world's particle physicists, come to CERN for their research. They represent 500 universities and over 80 nationalities.

Since its creation, CERN has made many important discoveries for which scientists have received prestigious awards, including Nobel prizes.

The one most useful for the public is the World Wide Web. It was developed to improve and speed-up the information sharing between physicists working in different universities and institutes all over the world[15], and now it has millions of academic and commercial users.

1.2 CDSware

1.2.1 Context

At CERN, research in particle physics and related areas outputs:

- 2,000 scientific publications per year;
- 10,000 conference talks and contributions per year;
- tons of experimental data.

As an international organization, CERN has been involved since its early beginnings with the open dissemination of scientific results. The dissemination

started by free paper distribution of preprints by CERN Library and continued electronically via FTP bulletin boards and the World Wide Web.

A digital library system for organizing and managing all the relevant information was the next logical step.

1.2.2 General overview

CDSware, or CERN Document Server Software as it is currently named is a complex software which provides the framework and tools for building and managing an autonomous digital library server. It is developed by, maintained by, and used at the CERN Document Server. At CERN, CDSware manages over 500 collections of data, consisting of over 800,000 bibliographic records, covering preprints, articles, books, journals, photographs and more. Besides CERN, CDSware is currently installed and in use by over a dozen scientific institutions worldwide.

Licensed under GNU General Public License (GPL), CDSware is thus a free open source software.

1.2.3 Information sharing

A scientist has to share his results, articles and preprints with his community. As search institutes are neither located in a single place, nor directed by a single team, various library systems have been developed. These systems possess differing capabilities, have distinct underlying bibliographic standards.

Electronic capabilities should be used to provide wide access to scholarship, encourage interdisciplinary research, and enhance interoperability and searchability.

Development of common standards will be particularly important in the electronic environment.

Principles for Emerging Systems of Scholarly Publishing
Tempe, Arizona, March 2-4, 2000

In order to facilitate the efficient dissemination of content, several unifying attempts have emerged. Nowadays, CDSware complies with Open Archives initiative metadata harvesting protocol (OAI-PMH) which enables metadata exchange between distinct library systems.

1.3 Social software

CDSware can be considered a mature and efficient search engine. At CERN, with about 20,000 unique visitors performing 200,000 searches per month, it has grown to one of the largest particle physics research digital library. Nevertheless, users could ask themselves why they should use a complex digital library system instead of their usual google-like search engine?

The aim of a search engine is to bring information to the user's fingertips, whereas the aim of a library is to provide means of advancing knowledge.

Information and knowledge are actually different concepts. Let's see what the wikipedia has to say about knowledge[6]



Knowledge = Theory + Information

As first defined by Richard L. Ballard, Ph.D. (1993), knowledge is the state of knowingness that results from the interaction between theory and information.

Theories are the rules, constraints, organization or conditional relationships between concepts, ideas and thought patterns that precisely define their meaning. Theory represents 85% of knowledge content. Learn once, use forever. People learn theory through enculturation, education and life experience. Well-justified theories have a useful life of decades to tens of thousands of years. Theory is *a priori*, known before the fact and gives information its meaning. Theory answers "How?" "Why?" and "What if?" questions.

Information represents anything that exists in time and space that can be processed by the senses, measured and counted. Information represents approximately 15% of knowledge content. Information answers "Who?" "What?" "When?" "Where?" and "How Much?" questions. Information is *a posteriori*, known after the fact.

The major difference between a digital library and a search engine is that a digital library allows complete bibliographic search (e.g author, date of publication, etc.). The second advantage is in direct relation with knowledge. A digital library should — and CDSware already offers it — provide collaborative features.

Knowledge cannot be stored as information. It can be obtained essentially by participating in discussions, experiments, research with people of the same community. Proposing collaborative features to users, thus helping users improve their knowledge is the aim of modern digital libraries.

1.4 Work

Although CDSware is a very complete system containing many extended features, the current personalization tools are under-employed. Only 2% of the users register themselves and access these tools. However, they seem convinced by these features, as 60% of the registered users use the current collaborative features (search baskets, alerts) already put in place. The main goal of this diploma project is to extend and enhance the collaborative features offered by CDSware and allow users to collaborate in more effective ways.

An analysis of user needs was already carried out by previous student, Paulo CABRAL[16]. As a result of this, partial implementation and design were carried out. The goal is to build on and extend his work, by implementing the core modules and refining the design.

1.4.1 Tasks

Learning

CDSware is mainly developed with Python 2.2. Training of Python is thus a prerequisite for the achievement of new functionalities.

As CDSware is a very complex system (about 250,000 lines of code), the implementation of new functionalities requires a global comprehension of the system. The discovering and understanding of CDSware's architecture, modules and coding styles has to take place at the beginning of this work.

WebMessage

In CDSware, due to policy rules, the email address of a user is never given to another. The side-effect of this policy is that, in order to use it collaboratively, users need an intern nickname and a messaging system. This feature should look like any other webmail system.

Implementation has to be done, with particular care in integration with other modules.

WebBasket

Users can save the bibliographic info of interesting documents in a basket. A basket should be either public, private or restricted for a group. Organization of private baskets by topics should be made available. A complete commenting system for entries in baskets should be possible. A system of access privileges (read, read comments, add comment, add article, etc.) must be developed.

A module already exists but it will have to be adapted/rewritten in order to make it possible to use it within groups of users. Export and import functions are currently developed by a CERN technical student and must be integrated in the new webbasket module. Design will be completed and the full implementation is required.

WebSession

If possible in time, the WebSession has to be refined to allow the creation and use of groups. Design is partially done.

1.5 Report organization

This report has been split in 5 chapters. The introduction chapter presents a general overview of digital libraries and an explanation of the social software concept. The second chapter introduces CDSware more precisely; a discussion about its capabilities, technical choices and architecture can be found there. The third chapter describes the first realization, WebMessage module. It completely discusses this new features. The fourth chapter is devoted to WebBasket module.

In chapter 5, reader will find general conclusions (module-related ones are found in chapters 3 and 4).

The bibliography and glossary can be found after chapter 5. Appendices take place at the end of this report and cover initial diagrams (made by an EPFL fellow[16]) and API details.

Chapter 2

Study of CDSware

This chapter discusses CDSware from a developer's perspective. It describes in particular its architecture and initial technical choices.

2.1 Users and access

CDSweb — the release of CDSware installed at CERN — is consulted by physicists all over the world. It is maintained by CERN librarians who validate submitted content and gather information from several search institutes. The validation process is a major difference between CDSware and a classical search engine.

CDSware has been designed in collaboration with the CERN Scientific Information Service (CERN Library), specifically for their needs. Almost every significant feature has to be validated by them. On the other hand, they give back a great wealth of information on user needs and problems.

2.2 Features

Developing new features for CDSware imply a strong integration with the current ones. Understanding the underlying technologies cannot be done if the developer doesn't know what CDSware looks like and what it is made for.

2.2.1 Search

The search engine permits to search through the bibliographic information (including documents stored on another digital library) and the full text of documents in CERN Document Server's (CDS) catalogues. Documents are sorted in collections according to their types. This search engine accepts convenient syntax, including boolean searches, regular expressions or field search (e.g. find documents where author is "John Ellis" and year of publication is 1985).

CERN Document Server

Home

Over **800,000** bibliographic records, including **360,000** fulltext documents, of interest to people working in particle physics and related areas. Covers preprints, articles, books, journals, photographs, and much more.

Search **812,077** records for:

any field

[Search Tips](#) :: [Advanced Search](#)

Narrow by collection:

- ☒ **Articles & Preprints** (661,235)
 - [Published Articles](#) (248,601) [Preprints](#) (338,097) [Theses](#) (35,903)
 - [Reports](#) (4,647) [CERN Internal Notes](#) (10,183) [Committee Documents](#) (26,056)
- ☒ **Books & Proceedings** (57,014)
 - [Books](#) (34,219) [Proceedings](#) (15,007) [Standards](#) (7,950)
- ☒ **Presentations & Talks** (15,018)
 - [Conference Announcements](#) (13,122) [Academic Training Lectures](#) (506) [Summer Student Lectures](#) (350) [General Talks](#) (1,367) [Videotapes](#) (299)
- ☒ **Periodicals & Progress Reports** (3,159)
 - [Periodicals](#) (2,475) [Progress Reports](#) (684)
- ☒ **Multimedia & Outreach** (27,968)
 - [Photos](#) (8,947) [Videos](#) (183) [Press](#) (15,495) [Audio Archives](#) (110)
 - [Exhibition Objects](#) (179) [Brochures](#) (9) [Posters](#) (325) [HEP Institutes](#) (924) [Experiments at CERN](#) (711) [Internet Resources](#) (1,092)
- ☒ **Archives** (53,121)
 - [CERN Archives](#) (47,036) [Pauli Archives](#) (3,711) [DSU Archives](#) (701) [SL Archives](#) (1,026) [AB Archives](#) (647)

Focus on:

- [CERN Articles & Preprints](#) (83,524)
- [CERN Published Articles](#) (44,109) [CERN Preprints](#) (10,758) [CERN Theses](#) (2,421) [CERN Reports](#) (196) [Committee Documents](#) (26,056)
- [CERN Departments](#) (59,207)
- [Accelerator Technology \(AT\)](#) (4,424) [Accelerators & Beams \(AB\)](#) (14,392)
- [Finance \(FI\)](#) (558) [Human Resources \(HR\)](#) [Information Technology \(IT\)](#) (1,999) [Physics \(PH\)](#) (34,700) [Secretariat-General \(SG\)](#) (6,027)
- [Technical Support \(TS\)](#) (1,035)
- [CERN Experiments](#) (13,071)
- [LEP Experiments](#) (4,927) [LHC Experiments](#) (8,148)
- [CERN R&D Projects](#) (39)
- [CERN Accelerator R&D Projects](#) (39)
- [CERN Series](#) (2,908)
- [CERN Yellow Reports](#) (1,088) [Academic Training Lectures](#) (506) [Summer Student Lectures](#) (350) [General Talks](#) (1,367)

Figure 2.1: Search interface

Due to a special in-house design CDSware, even with the massive number of documents boasts Google-like speeds with search times generally inferior to a half second.

CERN Document Server

Home > Search Results

Search: author

[Search Tips](#) :: [Advanced Search](#) :: [Try your search on...](#)

Search collections:

*** any collection ***

Sort by: **Display results:** **Output format:**

Results overview: Found **1,753** records in 0.39 seconds.

- [Articles & Preprints, 1,647 records found](#)
- [Books & Proceedings, 76 records found](#)
- [Presentations & Talks, 20 records found](#)
- [Multimedia & Outreach, 10 records found](#)

Articles & Preprints 1,647 records found 1 - 10

1. [Search for pentaquark states in Z decays / Schael, Set al.- ALEPH Collaboration.](#)
Exotic hadrons made of five quarks (pentaquarks) are searched for in hadronic Z decays collected by the ALEPH detector at LEP. [...] 2004 - Published in: [Phys. Lett., B 599 \(2004\) 1-16](#)
[Detailed record](#) - [Similar records](#)
2. [Constraints on anomalous QGCs in e⁺e⁻ interactions from 183 to 209 GeV / Heister, A et al.- ALEPH Collaboration.](#)
The acoplanar photon pairs produced in the reaction e⁺e⁻ → γγ are analysed in the 700 pb/sup -1/ of data collected by the ALEPH detector at centre-of-mass energies between 183 and 209 GeV. [...] 2004 - Published in: [Phys. Lett., B 602 \(2004\) 31-40](#)
[Detailed record](#) - [Similar records](#)

Figure 2.2: Search results appear at google-like speeds

2.2.2 Document submission

Authors can submit documents to CDS. In relation with types of documents being different on different installations of CDSware, the submitting page can vary a lot (the basic submission page can be seen on fig. 2.3). Eventually, submitted documents are passed through an approval mechanism.

The screenshot shows the 'textual document (Article, Preprint, Thesis, etc)' submission page. At the top is a header for 'ATLANTIS INSTITUTE OF FICTIVE SCIENCE' with navigation links like 'old.support@cern.ch', 'account', 'alerts', 'messages', 'baskets', 'approvals', 'administration', and 'logout'. Below the header are buttons for 'Search', 'Submit', 'Personalize', and 'Help'. The main heading is 'textual document (Article, Preprint, Thesis, etc)'. A sub-header reads 'textual document (Article, Preprint, Thesis, etc)' followed by a note: 'This is a template for a text type of document with direct integration after submission. Use it as a reference to create your new document types and submissions (close)'. Below this is a list of document types with radio buttons: 'Article', 'Book', 'Poetry', 'Preprint', 'Report', and 'Thesis'. To the right of these are three buttons: 'Submit New Record', 'Modify Record', and 'Submit New File'. A 'Notice:' section states: 'Select a category and then click the button to perform the action you chose.' At the bottom, there is a section for 'To continue an interrupted submission, enter your access number directly in the input box.' with an 'Access Number:' label, a text input field, and a 'go' button.

Figure 2.3: Submit documents

2.2.3 Current personalization features

Users can currently personalize CDSware with two features: alerts and baskets. Baskets will be discussed in more detail in chapter 4, as their enhancing is one of the goals of this project.

Alerts

Users can set an alert on his usual searches. He will be alerted periodically by email when a new document relevant to his search query is submitted. (e.g. “Alert me whenever John Ellis publishes a new article”, or “Alert me whenever there is a new article about Higgs bosons”).

Baskets

Users can store useful documents in a basket. This function is very similar to the eponym concept in a e-commerce.

2.3 Technologies

CDSware runs on GNU/Unix systems. Attempts are currently made to make it work on MacOSX, but this operating system is not yet supported. As CDSware

CERN Document Server

Home > Your Account > Set a new alert

Set a new alert

This alert will notify you each time/only if a new item satisfy the following query

QUERY: Pattern: ALEPH experiment: Candidate of Higgs boson production

Field: title

Collections: Articles & Preprints; Books & Proceedings; Presentations & Talks; Periodicals & Progress Reports; Multimedia & Outreach; Archives

Alert identification name:

Search-checking frequency:

Send notification e-mail? (if no you must specify a basket)

Store results in basket? or insert a new basket name

Figure 2.4: Setting an alert on a particular query

CERN Document Server

Home > Your Account > Your Baskets

Your Baskets

You own 1 baskets.

Select an existing basket: or

The selected basket is **Diplôme**.

Basket access is set to *private*, convert to *public*?

There isn't any alert related to this basket.

Selected items: or to

1 ☐ ▲ **ALEPH experiment: Candidate of Higgs boson production**
Expérience ALEPH: Candidat de la production d'un boson Higgs
 14 06 2000
 Keyword: [LEP](#)
 Photo number: CERN-EX-0106015

2 ☐ ▲ **Looking beyond the standard model / Ellis, J**
 CERN-TH.2004-028. Geneva : CERN, 2004. - mult. p
[Detailed record](#) - [Similar records](#)

Figure 2.5: Current basket interface

is an open source software, every underlying technology is also free.

2.3.1 Programming languages

Usual programming language is Python[12] 2.2.2. Many modules were initially developed using PHP[10] 4.3, work is currently ongoing to translate all the PHP into Python. Time-critical parts are written in Common LISP[4]. Compile-time configuration is made via GNU Autoconf[3] and WML[14] 2.0.8.

2.3.2 Web server

The web server is Apache[2]. 2.0.43 In order to integrate easily the programming languages, mod_python[7] 3.0 module and PHP module have been used. Templating is done through an in-house mechanism in order to be independant.

2.3.3 Database

The Database Management System (DBMS) is MySQL[9] 4.0. The database access is made through MySQLdb[8] 0.9.2.

2.3.4 The underlying bibliographic format

CDSware uses MARC 21 format. This standard is an edict of the Library of Congress (LOC), which is the official depository of United States publications.



The MARC formats are standards for the representation and communication of bibliographic and related information in machine-readable form.

<http://www.loc.gov/marc/>

As this format went out in the 1960's, it is based on outdated technology, but having such a long history guarantees a high level of persistence and flexibility, making this format very suitable for CDSware's use. Furthermore, the LOC has developed an XML version of MARC 21, much more convenient for modern systems.

2.3.5 OAi: Open Archive initiative

As said before, CDSware can share information with other digital libraries. This is done through a harvesting protocol: OAi-PMH[11]. CDSware is both a *Data Provider*, as it exposes metadata and a *Service Provider* as it uses metadata harvested from external sources. By default, OAi-PMH doesn't use MARC 21 XML format for data exchange, but Dublin Core format. Dublin Core, although being an XML format, offers far from MARC's granularity, as it only allows 15 metadata fields. However, the LOC has published means of converting from MARC to Dublin Core(via XSL transformation) and vice-versa.

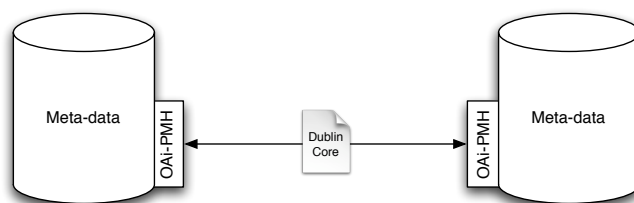


Figure 2.6: Exchange of metadata through OAi-PMH

2.3.6 Discussion

Technologies used in CDSware are not really usually encountered on the market.. Let us have a brief look whether they are well suited for their purpose.

Python advocacy

Choosing Python with mod_python Apache module instead of the most commonly used J2EE or .net may seem surprising. However, Python is a language with a lot of capabilities:

Easy to learn In an environment like CERN, teams of developers change often. Students come regularly for short periods, and must be able to develop as soon as possible.

Multi purpose language With Python, you can program following different paradigms: Object Oriented, Procedural or Functional. This makes this language very convenient.

Rapid prototyping Python is a real concise language — no need to put brackets, class definitions, etc. Moreover, the huge amount of embedded functions helps a lot in developing quickly complex functionalities. As Python is an “interpreted” language, the time-consuming compile-run cycles do not exist anymore.

However, Python lacks in several points:

Slowness Python is a bytecode-compiled language, thus it is much slower than compiled languages.

No standard There is no ANSI Python. This could become a problem in some years (some people currently want to drop useful functions).

Using Python when developing CDSware is really a good solution, as the time-critical parts are rare and the eventual standardization problem can be easily solved. Let’s see what a famous figure in the computing field thinks of it:

Python is an excellent language for my intended use. It is easy to use (interactive with no compile-link-load-run cycle), which is important for my pedagogical purposes. While Python doesn’t satisfy the prerequisite of being spelled J-A-V-A, Jython is close. Python seems to be easier to read than Lisp for someone with no experience in either language. The Python code I developed looks much more like the (independently developed) pseudo-code in the book than does the Lisp code. This is important, because some students were complaining that they had a hard time seeing how the pseudo-code in the book mapped into the online Lisp code (even though it seemed obvious to Lisp programmers).

The two main drawbacks of Python from my point of view are (1) there is very little compile-time error analysis and type declaration, even less than Lisp, and (2) execution time is much slower than Lisp, often by a factor of 10 (sometimes by 100 and sometimes by 1). Qualitatively, Python feels about the same speed as interpreted Lisp, but very noticeably slower than compiled Lisp. For this reason I wouldn't recommend Python for applications that are (or are likely to become over time) compute intensive. But my purpose is oriented towards pedagogy, not production, so this is less of an issue.

Peter Norvig, Director of Search Quality, Google

MySQL 4.0 advocacy

As the use of this version of MySQL may seem surprising, a historic look by CDSware's architect may help in understanding:

Initially, at around 1998, [developers] choose to use MySQL for a simple CDS application, because the inherent simplicity of the problem did not require usage of heavy and complex systems such as Oracle. It would have been an overkill. In the course of years, MySQL has proven very stable, scalable, and capable of dealing with very complex tasks. [18]

Unfortunately, as some users still run very old versions of CDSware, backwards compatibility impose to use this outdated version (no foreign key, no transaction, no trigger...). However, the next major version will undoubtedly set MySQL 5.0 as a standard.

An independant newspaper has tested several databases[13]. As a result, one can see that MySQL is a really scalable DBMS. Results can be seen on figure 2.7.

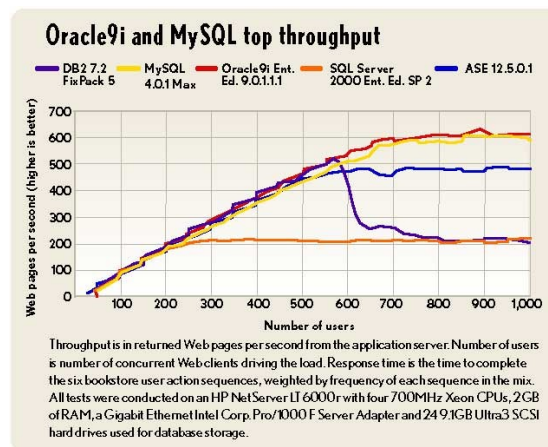


Figure 2.7: MySQL against other products with throughput[13]

2.4 Architecture

The general architecture of CDSware is a classic three-tier architecture. A browser is used to display infos, while the core functions are written (mostly) in Python and data is stored in a MySQL database. Interactions can be seen on figure 2.8. Please note that the database server can obviously be the same as the web application server. Actually, figure 2.8 depicts CDSweb installation.

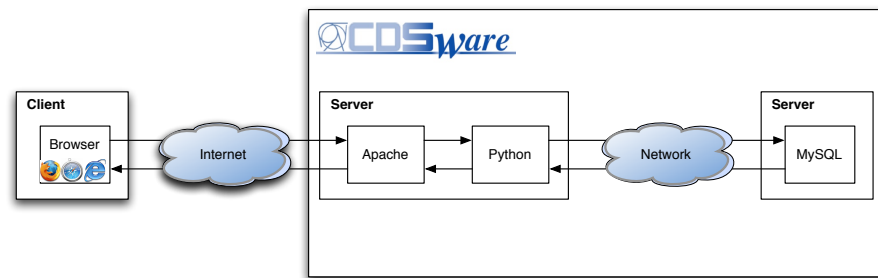


Figure 2.8: CDSware architecture

2.4.1 General organization

On an installed CDSware, administrators should find several directories. These are organized according to the well-known UNIX standards (fig. 2.9). The most important directory for an administrator is certainly the **bin** directory, which contains administrative functions (scheduler, harvester, etc.).

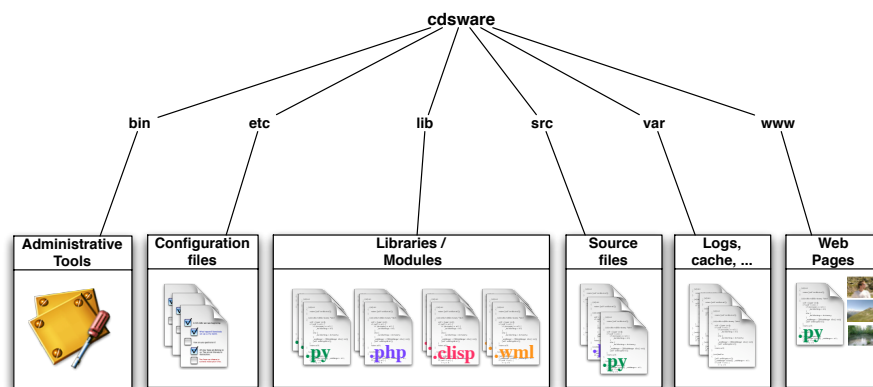


Figure 2.9: File organization in CDSware

2.4.2 Programming methodology

CDSware started as a set of relative independent modules developed by temporary people having independent styles. This was even more pronounced by the original use of many different languages (e.g. Python, PHP, Perl). Now the CDSware code base is striving to use Python everywhere, except in speed-critical parts as a compiled language (Common Lisp) could greatly enhance performances.

When building a big project like CDSware, using a unified methodology is mandatory. Here are the rules:

- All the python code should be extensively documented via docstrings, for use with `pydoc`;
- check code correctness and conform to coding standards using `pylint`;
- do not use magic constants;
- separate interface from core functions;
- name symbols descriptively (names as long as necessary);
- use imperative / functional paradigm rather than object oriented.

This last point has two main reasons:

1. Developers who instigated CDSware come from a background of functional programming,
2. Python object-oriented calls use 15-30% overhead.

Using a functional paradigm also helps avoiding over-engineering problems. The code is far more concise, and — highest priority of a search engine — runs much faster.

2.4.3 Modules

CDSware uses a modular design. This term does not mean Python module (equivalent to C libraries), but it signifies a component of the CDSware system. Modules contain core functions, interface functions and admin binaries.

A module is named according to the this rule: a module name is formed by prefix followed by the functional name. Prefixes can be “bib”, for bibliographic data related functionalities, or “web”, for interface related functionalities.

An overview of some useful modules and interaction between them is depicted in the next two pages.

Module overview

BibHarvest This module is responsible of harvesting and exporting metadata from and to fellow OAI-PMH compliant repositories.

ElmSubmit With this module, end users or non-OAI compliant libraries can also submit metadata, via a mail interface.

WebSubmit This module is responsible for the submission of documents via a web-interface. As in this case the full text document will be provided, this module also interacts with the file server.

BibConvert this module allows metadata conversion from any structured or semi-structured proprietary format into any other format, typically the MARC XML that is natively used in CDSware. Nowadays, BibConvert has been tested with data importation from over a hundred different data sources.

BibEdit This module enables a librarian to directly manipulate bibliographic data, edit a single record, do global replacements, and other cataloguing tasks.

BibCheck This module (currently in development) performs quality checks on metadata (e.g. a document published at CERN must have a department field).

BibUpload This module is responsible for loading metadata in the database. Data must be a well-formed MARC XML document. Usually these documents come from BibConvert module.

RefExtract This program tries to find out references in full text documents.

WebAlert This module is in charge of the alert personalization features.

WebBasket This module is in charge of the baskets personalization features. It will be discussed in more detail in chapter 4

WebSearch This module handles user requests to search for a certain word or phrase in the database. The system allows for complex boolean queries, regular expression searching, or a combined metadata, references and full text file searching in one go. Users have a possibility to browse for present index terms. If no direct match could have been found with the user-typed query pattern, the system proposes alternative matches as a search guidance

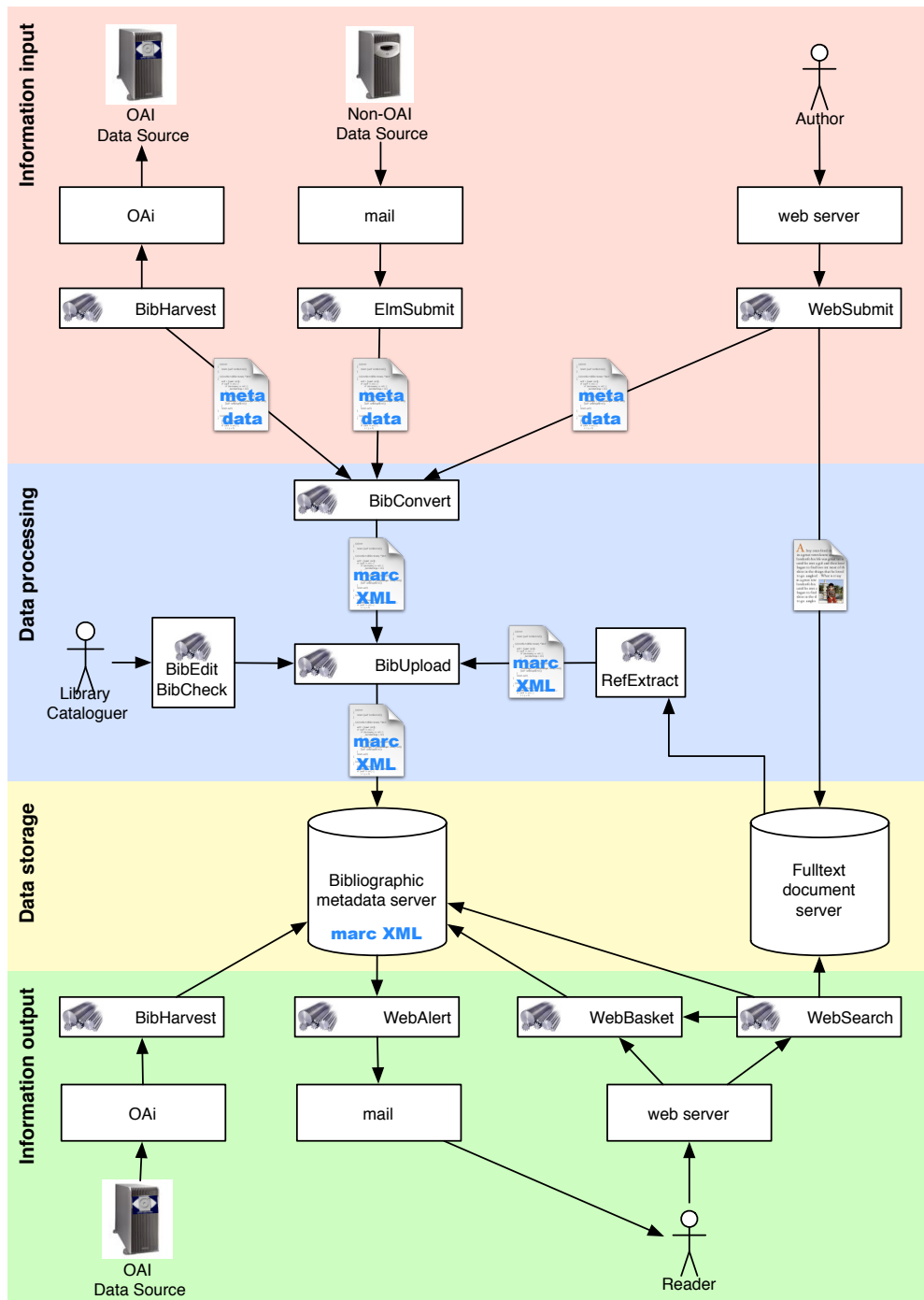


Figure 2.10: From input to output: module interaction

2.4.4 Operation principles

Each module should separate display functions and business logic. On fig. 2.11, one can see a sequence diagram of a typical page generation. The model used here is the search engine. However, in reality the WebSearch module doesn't exactly comply with this diagram, as it progressively outputs results in order to improve response time. All other modules *should* respect this way of working.

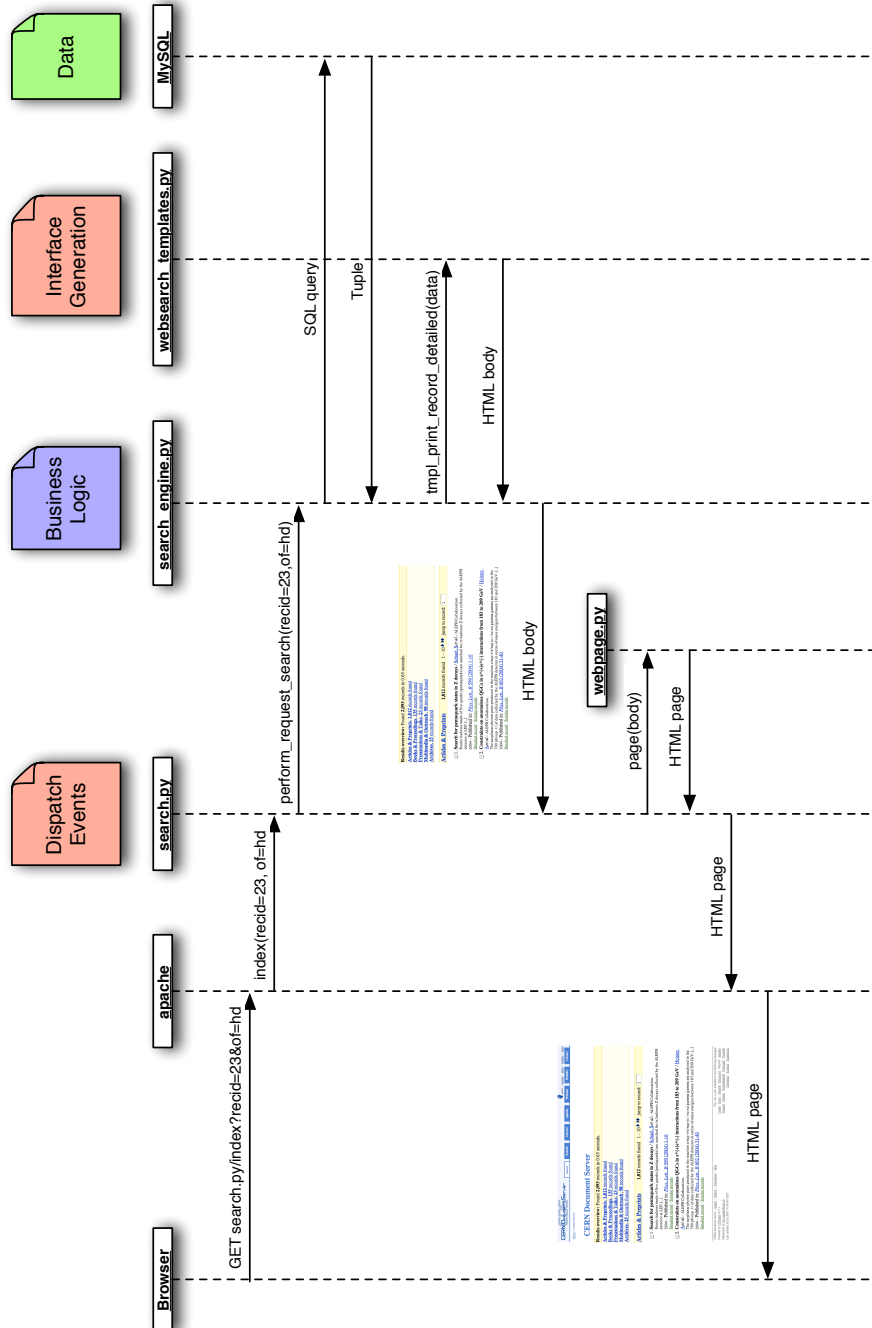


Figure 2.11: Sequence diagram of a page generation

Chapter 3

WebMessage

WebMessage module aim to provide a messaging system. Users must indeed be able to communicate privately. Because the user decides which groups he is part of, such a system would also be convenient to invite other users to join a group, etc.

The problem is that they cannot use their usual email address. Actually, the policy in this software forbids the providing of personal informations.

3.1 Specifications

An analysis of user needs has already been realized by Paulo CABRAL[16] for this part of the collaborative features. Discussing with librarians and development team, he carried out paper prototypes of the new features and a database schema (see appendix B.1). The first stage will be a validation of his design work. Thereafter, the following functionalities will be implemented:

- Display, write, send messages and reminders (a message sent on a further date) to users or groups;
- search for users or groups;
- create a command line interface (CLI) for administrative tasks (clean of database, etc.).

3.2 Analysis

Paulo Cabral did certainly a very good job when discussing with librarians and developers. His analysis is very complete and defines with a high level of precision the specifications of the modules. However, a refining work needs to be done.

3.2.1 User interface

Prototypes of the user interface have been received at the beginning of this work. Prototypes relevant to discussion will be displayed here. All other ones can be found in appendix B.2, page 61.

Users of CDSware are usually accustomed to online messaging systems such as gmail. This module's intent is to look like any webmail service in order to decrease training time.

User Guidance

The user interface sketches, while being finely detailed, lacked in user guidance. Users — especially non IT ones — must be guided through the whole process. Users can indeed easily enter into difficulties:

1. recipients aren't mentioned or do not exist;
2. date of reminder is false (e.g. 30th of February),
3. recipients are over-quota

A generic method of displaying errors or informations has been found.

Search box

A way of searching users is depicted on the sketches [fig. 3.1]. The way it is done isn't very clear or usable. As thousands of people currently use CDSweb, imagine a "find user" box where every name is displayed!

Figure 3.1: Search users or groups

A real search option must be outlined. User should give a pattern and press a classical search button. Results could then be selected and added to the list of recipients.

Using two distinct interfaces for basically doing the same thing isn't very user friendly. A single search interface, with a "Search for users" and a "Search for groups" buttons is preferred. The "Add" buttons are then to be renamed correctly, as the system knows if it has performed a user or group search. With that way of thinking, lists can also be long, but it would be the user's will (e.g. search of pattern "e").

3.2.2 Database

A diagram of tables was carried out in the initially received analysis and covers the whole collaborative features. but it contains too much relationships for the discussion of the WebMessage module. Only messaging related parts will be discussed here. The complete initial diagram, however, can be found in appendix B.1, page 60.

As one can see on figure 3.2, this diagram isn't a proper relational or Entity-Relationship one. In fact, it's a strange mix of diagrams. As the syntax very informal, it has to be completely redesigned.

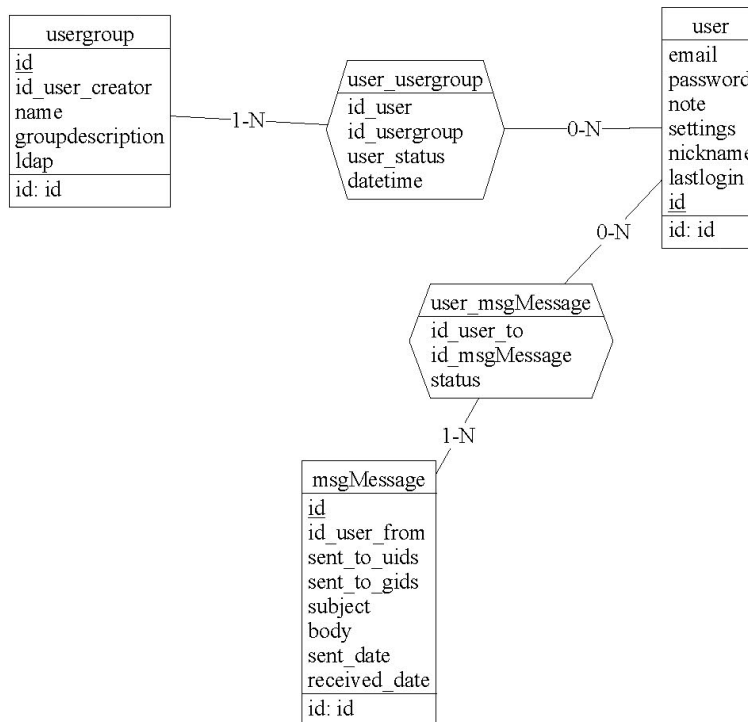


Figure 3.2: Initial incorrect diagram

Entity-Relationship diagram

After several discussions with CDSware’s architect, Tibor Šimko, some refinements were carried out.

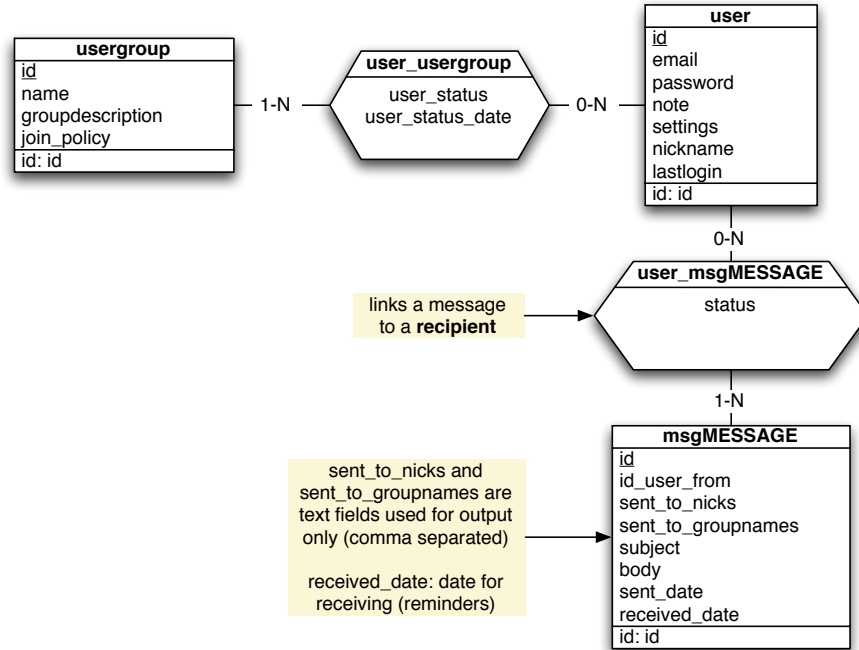


Figure 3.3: Entity Relationship diagram

User groups Naming of fields isn’t very explicit. As CDSware uses a relative complex database (more than 300 tables), naming has been improved.

The originally mentioned LDAP field won’t be used since administrative utilities will be developed for the LDAP linking. On the other hand, a “join policy” field will be useful.

Messages A field is initially used to store a list of recipients (`sent_to_uids`). Looking closely at documentation and code, it was found out that this field would only be used in interface-related parts. This field, as being textual, should not contain ids, but nicknames. Thus it has been renamed to `sent_to_nicks`, more representative of what it should contain.

Relational diagram

From the ER diagram, a relational one was carried out (figure 3.4).

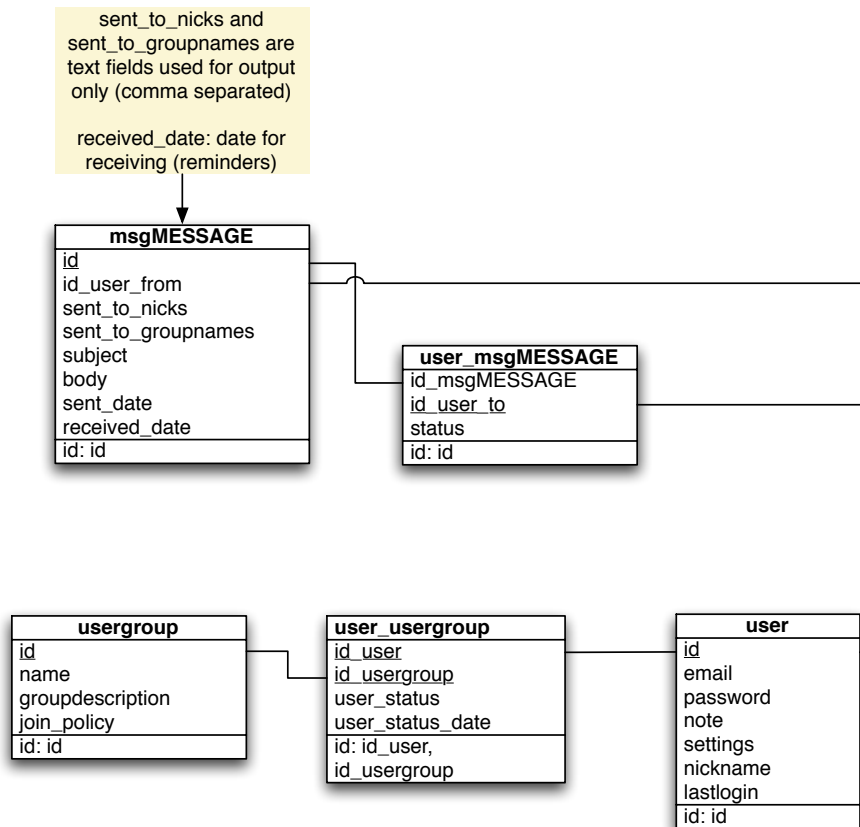


Figure 3.4: Relational diagram

Nickname Guest users are allowed and logged in table `user`, but do not possess a nickname. Therefore, the nickname field can be null in this table. In fact, this field was added to the previous version of CDSware in order to properly handle comments. It will be very convenient to use nicknames to send a message. This could lead to a simple modification of the WebComment module for the integration of the messaging system.

sent_to_nicks This field and also `sent_to_groupnames` field are text based. They won't be used for anything but display. Its existence is due to the will of not creating another table, avoiding complicated join queries.

3.2.3 Architecture

According to CDSware’s coding standards, interface has to be separated from the business logic. As the formal architecture leads to very long files, a decoupling choice has been made: The database-oriented functions will be located in another file. This way of working also facilitates updates.

As object-oriented development is “prohibited” in CDSware, class diagrams do not make a lot of sense. Instead, sort of collaboration diagrams have been used. The following diagram (fig. 3.5) shows interaction between python modules (files).

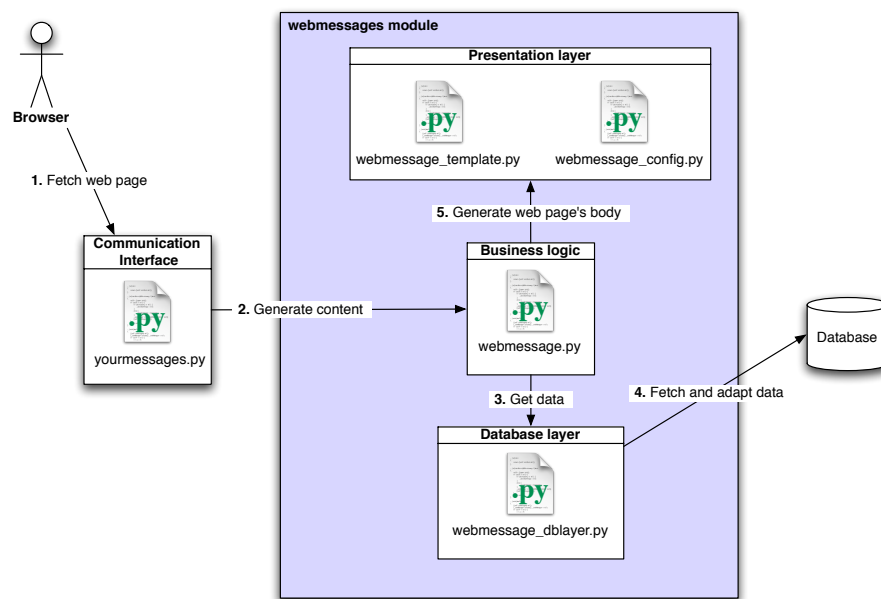


Figure 3.5: WebMessage: collaboration diagram

yourmessages.py This python module is responsible for the interaction with user. Thus it will be deployed in the `www` directory (see figure 2.9, page 14 for a reference on directory structure).

webmessage.py This file contains the business logic. Modules that will use WebMessage will interact with this python module. It will be deployed in the `lib` directory

webmessage_dblayer.py This python module knows how data is structured in database. As it doesn’t know anything about the business logic, it only provides tools for data access, and won’t verify user rights. It will be deployed in the `lib` directory

webmessage_templates This file is responsible for HTML rendering of messages. It only knows how to display results and will be deployed in the `lib` directory.

webmessage_config This python module exists to avoid magic numbers or strings. It knows what text to display in case of errors and specific configuration (e.g. quota, maximum length of messages, etc.). It also contains dictionaries for data insertion in the database. An example is cited below.



```
# status of message (table user_msgMESSAGE)
cfg_webmessage_status_code = \
{
    'NEW': 'N',
    'READ': 'R',
    'REMINDER': 'M'
}
```

3.3 Realization

The complete implementation of this modules was carried out in three development stages. The first one concluded with a complete usable interface. The second one was devoted to scalability tests, optimization and enhancing of the code. The third stage was a complete refining of the implemented module, as user / developer feedback was available.

3.3.1 Database

Using MySQL 4.0 with MyISAM storage engine prevents use of modern database tools. Useful concepts like foreign keys, stored procedures, subqueries, triggers or transactions. Therefore, these mechanisms have to be implemented in business logic.

Transactions

Some transactions have been identified. Neither transactions nor effective locking functions are implemented in MySQL MyISAM. An answer to this lack has been addressed by development of administrative tools. For most of transactions, errors won't cause any major problem like database inconsistency, and so choice has been made not to care about them.

The sending of messages is a potential creator of inconsistencies. In fact, the sending process is formed by two phases:

1. Creation of a message in table `msgMESSAGE`,
2. Sending of the message via multiple insert statements in the linking table `user_msgMESSAGE`

If the system crashes (e.g. power failure) between the two phases or in the last phase (in this case, MySQL engine will rollback the last inserts), a message would be created with no recipients. If nothing were done, the database would grow with no mean of purging it. The same problem happens if the system crashes while users destroy messages.

As the user management is the responsibility of another module, in case of a user suppression, database also needs to be cleaned. The intention of the developed administrative library is to provide a complete cleaning solution.

For these reasons an administrative tool for the database cleaning has been developed. As this piece of software is a CLI, it can be automatically run in a batch mode started periodically by CDSware's scheduler.

3.3.2 Scalability tests

The scalability tests were done on the most time-consuming functionality. The displaying of inbox or messages couldn't take a lot of time because of their inherent simplicity. On the other hand, the sending of a message could take a lot of time as its algorithm requires several database queries and potentially a lot of loops:



```

check user input
get user IDs from nicknames
add user IDs to list of recipients
get group IDs from groupnames
get IDs of users members of groups
for all user IDs members of groups do
    if user ID not in list of recipients then
        add user ID to list of recipients
    end if
end for
create message
for all user IDs in list of recipients do
    send message to user
end for

```

After several tests, it was found out that the recovery of IDs from names and the final sending were the most time-consuming processes. The first idea for the resolution of this problem was to use dictionaries in order to decrease the number of queries as this could suppress useless ones. Then, putting invariants out of loops could also save some time. These solutions divided execution time by a factor of 4. This wasn't sufficient at all!

A big point of optimization had to be found. What had this functions in common? They did as much queries as there were users to send message to. This led to a potentially high number of queries, which is very slow. A way of understanding this can be found on MySQL website:



Speed of INSERT Statements The time required for inserting a record is determined by the following factors, where the numbers indicate approximate proportions:

- Connecting: (3)
- Sending query to server: (2)
- Parsing query: (2)
- Inserting record: (1 x size of record)
- Inserting indexes: (1 x number of indexes)
- Closing: (1)

<http://dev.mysql.com/doc/refman/4.1/en/insert-speed.html>

While connection and closing times can be neglected as this process occurs only once in a page generation, the sending and parsing queries processes were done very often. The solution was to write very long queries instead of a lot of short queries. Results were very impressive:

Number of recipients	Short queries [s]	Long query [s]
10	0.611843	0.00059
100	4.603257	0.001065
1,000	105.821297	0.040322
10,000	217.616954	3.456474
100,000	> 2 hours	488.81623

This is even more explicit on a graph:

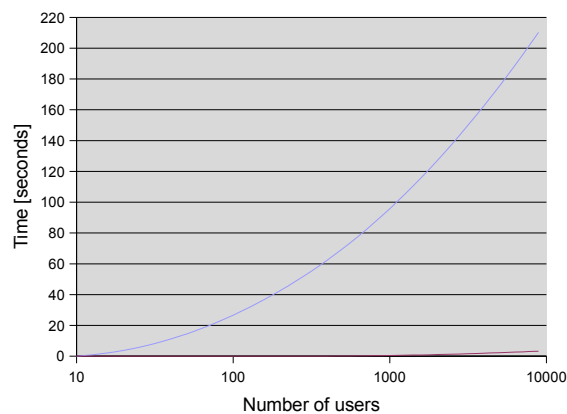


Figure 3.6: Old versus new version of sending algorithm

Tests were done on a machine far less powerful than CDSweb server, with conditions up to 1,000 times harder than the expected use.

Tests were also done on complex queries. Under expected circumstances — 10,000 users possessing 50,000 messages — there is no query exceeding a fifth of second.

One can see that, due to the obtained results, WebMessage module is now more than capable of handling expected load.

3.3.3 User interface

The user interface, as defined on M. Cabral’s work, was improved. It was also tested by other members of CDSware development team who helped making it clearer. The search box, as asked in specifications works finely in a comfortable way.

One of the prerequisites was to provide correct HTML. The model choosed by CDSware’s architect is HTML 4.0.1 transitional[5]. Every part of the interface has been validated through the W3C’s validator¹

Error messages are now displayed by a box on the top of window. Helping with user guidance, information messages such as “Message Sent” or “Message deleted” were added.

Usability has been improved with direct links to functions (delete, reply, etc.).

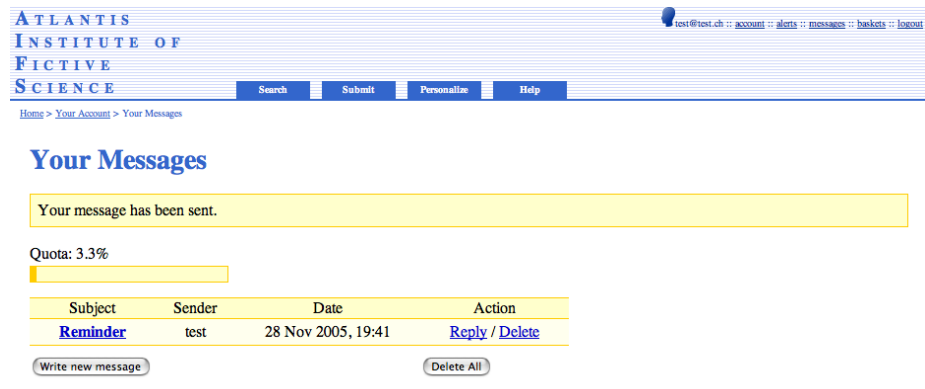


Figure 3.7: A user’s inbox, webmessage’s entry point

¹See <http://validator.w3.org>

Write a message

Warning:
User 'georges' doesn't exist

To: Users

Groups

Subject:

Message:

Send Later:

Find users or groups

Figure 3.8: Error messages are displayed

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

test@test.ch :: account :: alerts :: messages :: baskets :: logout

Search Submit Personalize Help

Home > Your Account > Your Messages > Write a message

Write a message

To: Users

Groups

Subject:

Message:

Send Later:

Find users or groups

Geeks usergroup
group 2
group 3

Atlantis Institute of Fictive Science :: Search :: Submit :: Personalize :: Help
Powered by CDS.ware v0.7.1.20051108
Maintained by cds.support@cern.ch
Last updated: \$Date: 2005/11/14 10:41:24 \$

This site is also available in the following languages:
Català Češky Deutsch Ελληνικά English Español Français Italiano
Norsk/Bokmål Português Pycckий Slovensky Svenska Ҳиндустани

Figure 3.9: Users can search for others

3.4 Conclusion

The realization of this first module was successfully carried out and validated.

Publication The WebMessage module, and other global modifications have been published on the developer's CVS.

Official release The WebMessage module will be announced, with the other collaborative features in the next official release of CDSware, which is expected for mid February 2006.

Development of WebMessage also helped improvement of the whole project:

HTML validation Testing HTML output, with the help of the developped stylesheets on several configurations, led to the correcting of an Internet Explorer specific problem in the display of the WebSearch module. Lot of time was spent in finding a generic solution. Now, the whole project works fine on every usual browser (tested on Internet Explorer, Firefox, Safari, Konqueror, Opera and Lynx).

Furthermore some small HTML inconsistencies prevented validation of pages. This was corrected, permitting validation of every page of CDSware.

Date handling API As WebMessage module was developed, a coherent way of displaying dates was found to be necessary. A new library of the MiscUtils module has been implemented. It defines the way the dates are to be used in CDSware (display, intern, and database). This library is fully internationalized. It also permits HTML output of several date selection fields, helping decrease global code length.

Persistence layer separation The separation of database functions from business logic is now a standard for the development in CDSware.

This module has been tested on every function it provides, especially on edge effects. It seems not to contain any bug and therefore reaches the general level of excellence of CDSware.

Chapter 4

WebBasket

Currently, in CDSware, users save particular bibliographic data in baskets. This system, while being appreciated only allows private or public baskets, and dot not allow topics. In order to help collaboration between groups of users, this system has been entirely rethought.

4.1 Specifications

An analysis of user needs has already been realized by P. CABRAL[16] for this part of social tools. Prototypes have been carried out by the discussion with librarians and developers. The first stage will be a validation of his design. The following features need to be implemented:

- organization of baskets in categories;
- creation of group baskets with access privileges;
- commenting on basket's items;
- importation of external ressources (with a given URL or BibT_EX file);
- exportation of a baskets' content to BibT_EX format.

4.2 Analysis

A work of refinement, especially on the database needs to be done over the previous student's work.

4.2.1 User interface

Detailed prototypes of the forthcoming interface have been received in the beginning of the project and can be found in appendix B.3, page 66.

This system has to be easy to use, as users aren't expected to be accustomed to this kind of service. Special effort has to be made for guidance part of Human-Computer Interface (HCI).

User guidance

The major problem on the prototypes is the lack of helpful informations. As an example, let's see how people can edit a basket:

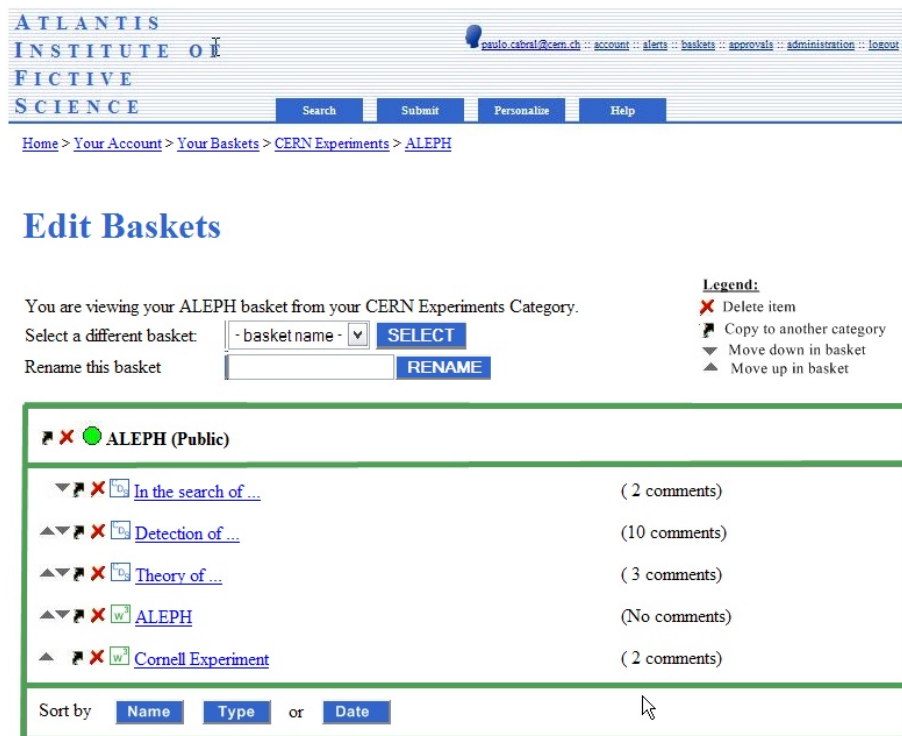


Figure 4.1: Editing of a basket

This screenshot isn't clear. No information seems to be provided about basket sharing — although it's the intent of the green dot. As baskets can be shared, an interesting information would be the display of dates of last added item or last comment.

4.2.2 Database

The part of database diagram covering baskets which has been provided at the early stage of this work (fig. 4.2) suffers the same formalism problems as discussed earlier in WebMessage's analysis. Moreover this diagram doesn't cover correctly all the specifications (which have mostly been carried out by its creator). For these reasons the diagram has to be rewritten from scratch.

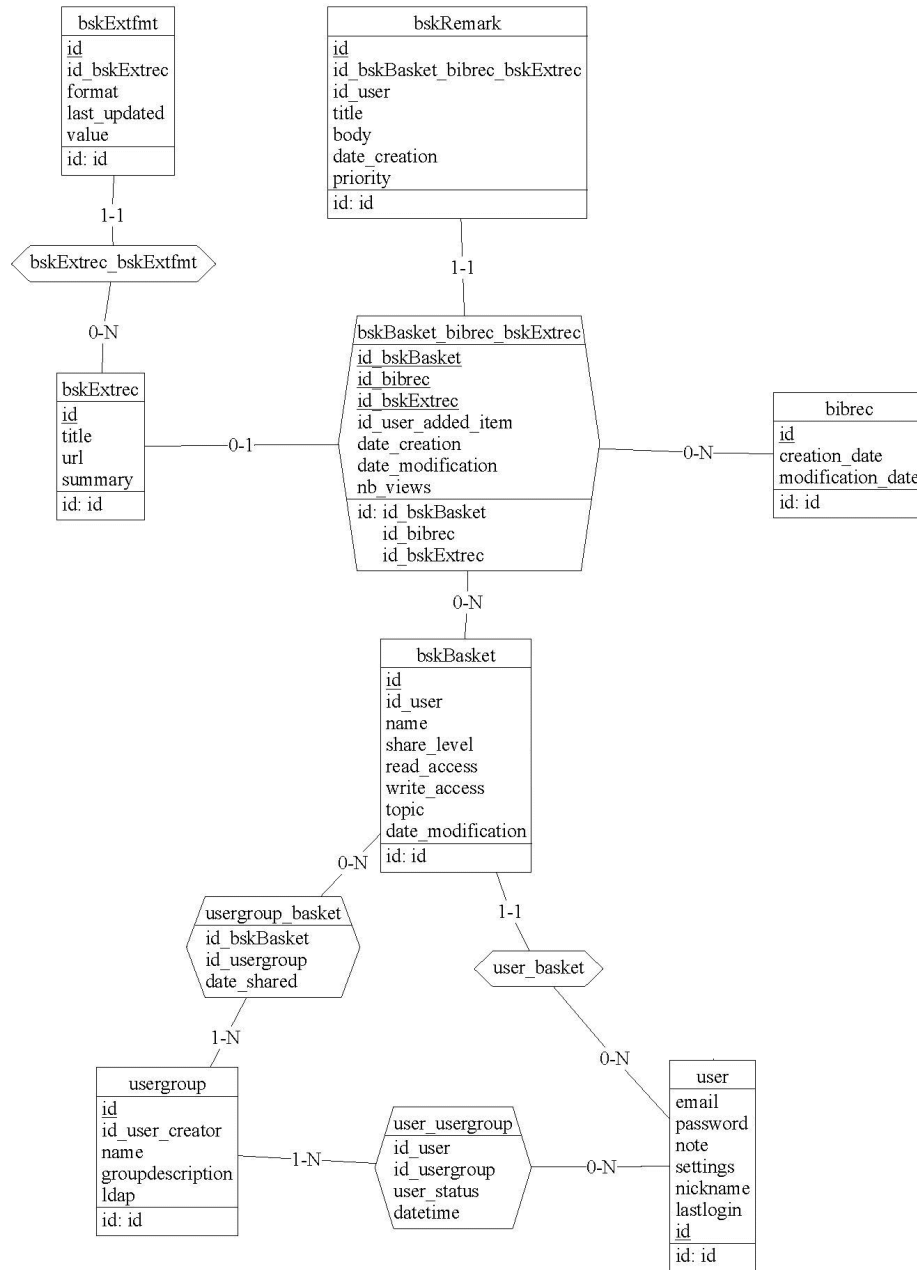


Figure 4.2: Initial informal diagram as received from [16]

The interactions have been separated into two views for better understanding. Complete diagrams can be found in appendix C

Baskets interaction with users and groups

After several discussions with CDSware’s architect, with respect to the relation between users and groups defined in WebMessage’s analysis, the following (figure 4.3 Entity-Relationship diagram was carried out.

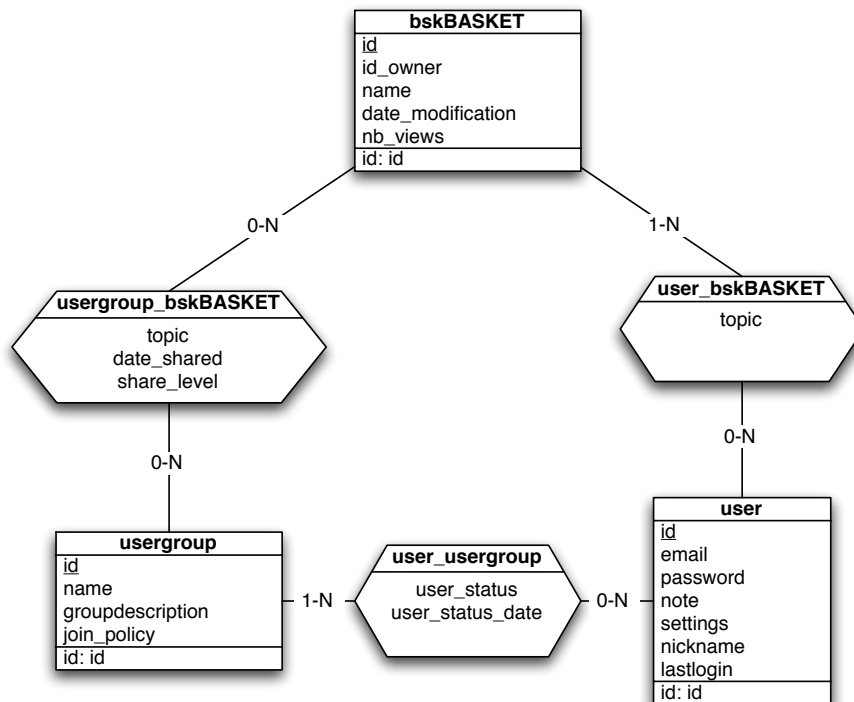


Figure 4.3: WebBasket’s Entity-Relationship diagram, part 1

Topics Users can create as much topics as number of baskets they subscribed to. A topic field is also mentioned in the relation between groups and baskets and is reserved for a future use.

Access rights The initial model has been simplified. A basket can only be shared with groups as one can read on the relationship **usergroup_bskBASKET**. A special group is used for the “anybody” access. The share level can contain any rights for reading comments, posting comments, adding new items, etc.

Relational diagram From the ER diagram, a relational one was carried out (fig. 4.4)

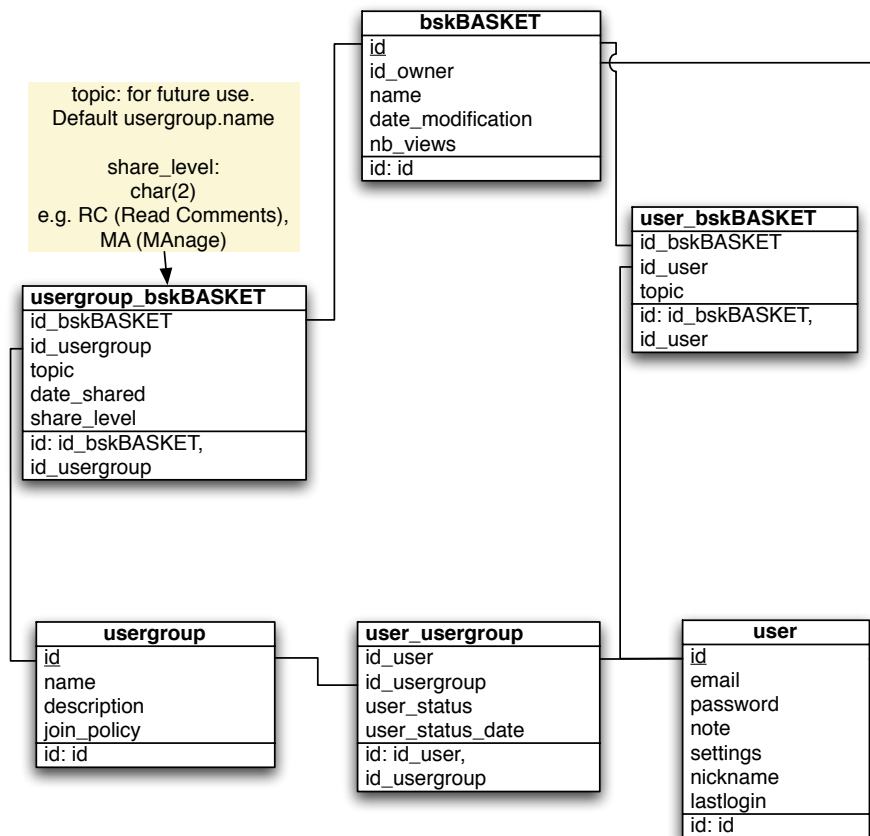


Figure 4.4: WebBasket's Relational diagram, part 1

Share_level A choice has been made to use chars rather than anything else. This permits the creation of a new rights type easily, and is convenient for the programmer. This philosophy is used at several places in CDSware's database.

Baskets interaction with records and comments

Two types of records will exist in CDSware:

1. Internal records (already existing);
2. External records.

Internal records entity cannot change, while it is used by almost every module in CDSware. External records entity has to be defined the same way for consistency reasons. As internal records entity cannot be altered in implementation, specialization won't be used in this diagram (fig. 4.5).

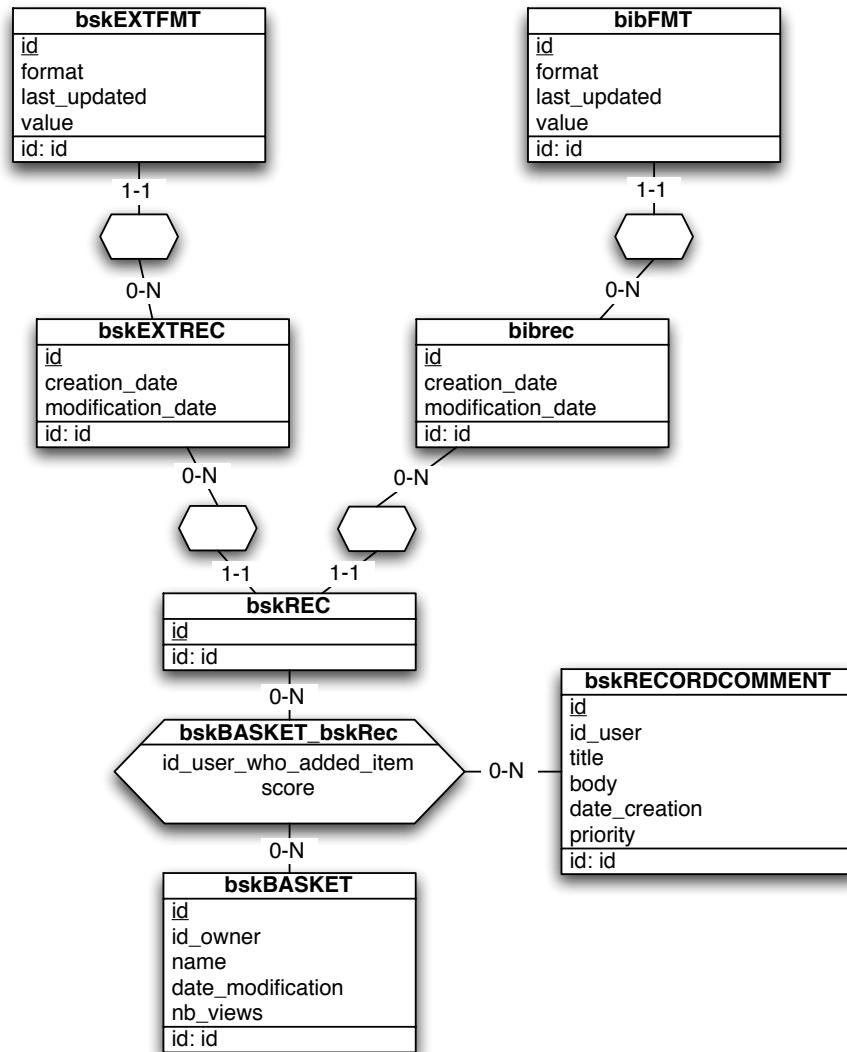


Figure 4.5: WebBasket's Entity-Relationship diagram, part 2

Comments Each record in a basket can possess comments. These comments have a priority, reserved for a later use. The score used in the relation between baskets and records is reserved for sorting purpose.

Records and formats Each record is linked to various formats. A format can be HTML Brief, HTML Detailed, MARC 21, etc. Formats represent bibliographic information in the respective context.

Relational diagram From the previous ER diagram, a relational one has been developed (fig. 4.6). As this diagram has been severely denormalized, it may not seem correct. However it has been submitted to and approved by CDSware's architect.

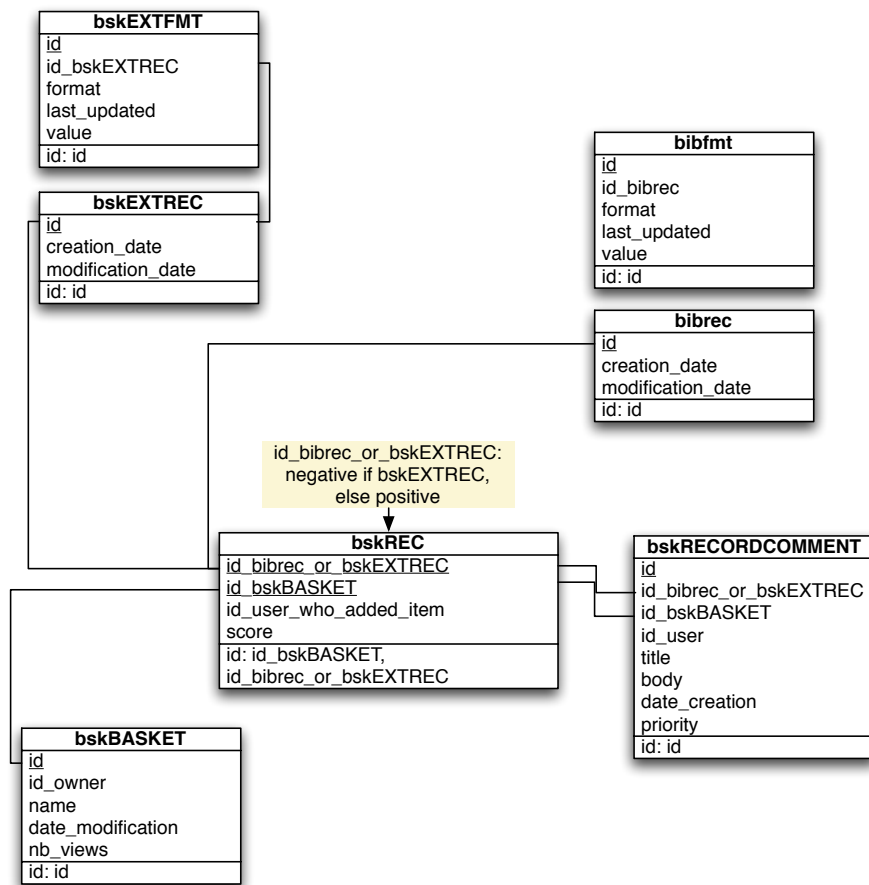


Figure 4.6: WebBasket's Relational diagram, part 2

Records As said before, a record can be either internal or external. The direct relationship between records and baskets has been abandoned, as it wouldn't be useful. A surprising choice has been made with the primary key of a record: This key is either positive (case of internal record), or negative (case of external record). The number following the sign represents id of internal or external record. As MySQL MyISAM doesn't provide a lot of modern tools, this was

certainly the best solution, and is relatively common in CDSware. The main drawback of this method is that the business logic will have to handle this potential creator of inconsistencies.

4.2.3 Architecture

As in WebMessage module, interface has been separated from the business logic. The decoupling of database related parts from business logic will also be used in this part. This will help in solving the problem mentioned in database analysis. The following diagram (fig. 4.7) is, as in WebMessage module, a collaboration diagram.

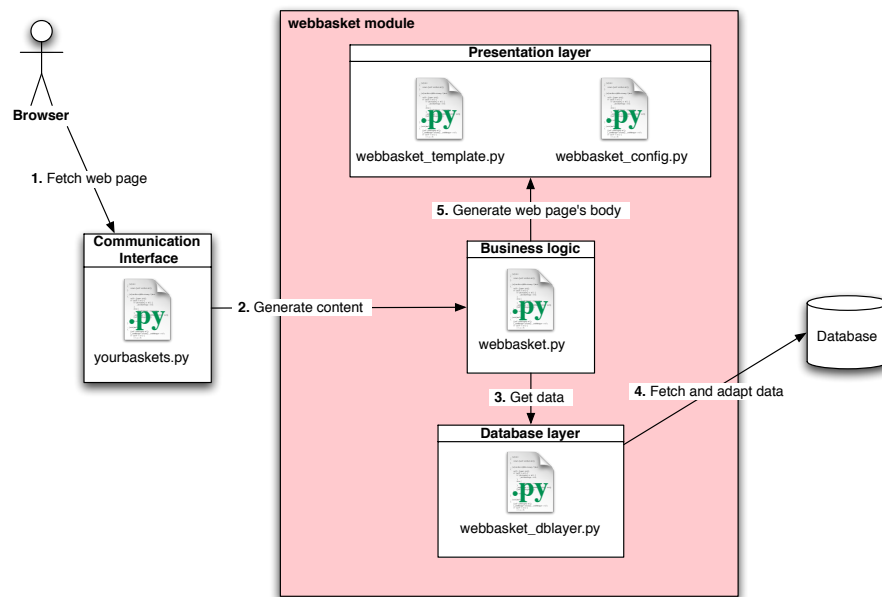


Figure 4.7: WebBasket's Architecture

yourbaskets.py This python module is responsible for the interaction with user. Thus it will be deployed in the `www` directory (see figure 2.9, page 14 for a reference on directory structure).

webbasket.py This file contains the business logic. Modules that will use WebBasket will interact with this python module. It will be deployed in the `lib` directory

webbasket_dblayer.py This python module knows how data is structured in database. As it doesn't know anything about the business logic, it only provides tools for data access, and won't verify user rights. It will be deployed in the `lib` directory

webbasket_templates This file is responsible for HTML rendering of baskets. It only knows how to display results. It will be deployed in the `lib` directory.

webbasket_config This python module exists to avoid magic numbers or strings. It knows what text to display in case of errors and specific configuration. It also contains dictionaries for data insertion in the database. An example is cited below.



```
# access rights of a basket(table usergroup_bskBASKET)
cfg_webbasket_access_rights = {'READITM': 'RI',
                              'READCMT': 'RC',
                              'ADDCMT': 'AC',
                              'ADDITM': 'AI',
                              'DELCMT': 'DC',
                              'DELITM': 'DI',
                              'MANAGE': 'MA'
                              }
```

4.3 Implementation

The realization process is well advanced but not finished at current time. However, the database implementation is fully implemented and has proven efficient by the first coding process.

4.3.1 Database

Transactions and eventual cleaning functions have not been developed as it usually takes place after the complete development process when database schema may have to be adapted. However every display related queries have been created. The experience acquired in the creation of WebMessage certainly helped in development of better and faster queries.

4.3.2 User interface

The display of baskets, with group baskets and categories has been fully implemented. As discussed before a special effort has been made on visualization of sharing, dates, etc. Thus, icons had to be added. As CDSware is open source, icons also have to be free. It was decided to use and modify icons from Gnome, a widely known desktop environment.

The set of icons should represent user, groups and public sharing of groups. The icons in figure 4.8 were chosen.

The final interface is very simple. Regarding to rights, functions are or are not displayed. This can be seen on figures 4.9 and 4.10.



Figure 4.8: Sharing icons

ATLANTIS INSTITUTE OF FICTIVE SCIENCE

cds.support@cern.ch :: [account](#) :: [alerts](#) :: [messages](#) :: [baskets](#) :: [approvals](#) :: [administration](#) :: [logout](#)

[Search](#) [Submit](#) [Personalize](#) [Help](#)

[Home](#) > [Your Account](#) > Your Baskets

Your Baskets

Personal Basket Science Geeks usergroup group 2 group 3 Other's baskets

Admins personal basket
0 records - last update: 05 Nov 2005, 00:00 0 comments
last comment:

Sort by: [Title](#) [Date](#)

Atlantis Institute of Fictive Science :: [Search](#) :: [Submit](#) :: [Personalize](#) :: [Help](#)
Powered by CDSweb v0.7.1.20051115
Maintained by [cds.support@cern.ch](#)
Last updated: \$Date: 2005/05/13 08:41:16 \$

This site is also available in the following languages:
[Català](#) [Česky](#) [Deutsch](#) [Ελληνικά](#) [English](#) [Español](#) [Français](#) [Italiano](#)
[Norsk/Bokmål](#) [Português](#) [Русский](#) [Slovensky](#) [Svenska](#) [Українська](#)

Figure 4.9: A group shared basket, manager rights

ATLANTIS INSTITUTE OF FICTIVE SCIENCE

cds.support@cern.ch :: [account](#) :: [alerts](#) :: [messages](#) :: [baskets](#) :: [approvals](#) :: [administration](#) :: [logout](#)

[Search](#) [Submit](#) [Personalize](#) [Help](#)

[Home](#) > [Your Account](#) > Your Baskets

Your Baskets

Personal Basket Science Geeks usergroup group 2 group 3 Others' baskets

a test basket
5 records - last update: 25 Dec 2004, 00:00 3 comments
last comment: 26 Oct 2005, 20:00

ALEPH experiment: Candidate of Higgs boson production
Expérience ALEPH: Candidat de la production d'un boson Higgs
14 06 2000

Abstract: Candidate for the associated production of the Higgs boson and Z boson.
[...]

Keyword: [LEP](#)
Picture number: CERN-EX-0106015

2 comments; last: 25 Oct 2005, 20:00
[Details and comments](#)

1 comments; last: 26 Oct 2005, 20:00
[Details and comments](#)

0 comments; last: 00 Month 2000, 00:00
[Details and comments](#)

© CERN Geneva

Figure 4.10: Someone else's basket, commenting rights

4.4 Conclusion

The design of the database took a lot more time than initially thought. This lost of time was essentially due to the complete rewriting of the diagrams. Now that it has been approved and verified, development can go really faster.

An estimate of 3 weeks/man is necessary to complete and test this module.

Chapter 5

General conclusions

5.1 Results

The development of new social software modules for CDSware permitted a complete walk through the development cycle. Learning of development environment, researching on users' needs, designing of technical documents, implementation of new functionalities and testing every aspect of developed software were ambitious tasks. Here's an overview of obtained results:

Objective	Results
Learning of CDSware's environment	fulfilled
Designing of WebMessage module	100%
Implementation of WebMessage module	100%
Testing of functionalities of WebMessage module	100%
Testing of scalability of WebMessage module	100%
Designing of a WebBasket module	80%
Implementation of WebMessage module	50%
Testing of functionalities of WebMessage module	50%
Testing of scalability of WebMessage module	0%
Enhancements of WebSession module (optional)	Not carried out

As one can see, the WebBasket module didn't come to an end. This is essentially due to the will of perfecting WebMessage. In fact this module was finished since two weeks when user feedback was returned. The deliberate choice was made to stop WebBasket's development and rather improve WebMessage to the perfection.

Development of these modules also led to a global improvement of CDSware, by addition of new libraries and correction of bugs:

- date handling API for the parsing and display of dates;

- text handling API for the output and display of text (quoting and indenting);
- global CSS redefinition, correcting and enhancing major CDSware features;
- perfection of global HTML output to fulfill W3C's norm HTML 4.0.1 Transitional;
- tests of incompatibilities between Python 2.2 / 2.3 and MySQL DB connector 0.9 and 1.2. This tests will permit the complete upgrade of the versions.

5.2 Improvements

The next step would obviously be the completing of WebBasket and the enhancement of WebSession modules. As the next official release is expected for middle of February 2006, this development should take place as soon as possible.

Objective	Duration
Finishing implementation of WebBasket module	3 weeks
Tests / validation with user feedback of WebBasket module	1 week
Analysis and design refinement of WebSession module	3 days
Implementation of WebSession group handling	1 week
Tests / validation of WebSession group handling	1 week

5.3 Personal results

Having worked in CERN was certainly the greatest achievement of my career. Having pleasure of getting up and being proud of what I did and for who I did it was just fantastic!

While every part of this work did not come to the end I wanted to, the results were here and really satisfying. An opportunity of staying at CERN some more weeks, continuing this work was offered and, of course, I accepted!

Acknowledgments

Although this report is written in English, the following acknowledgments are written in French — my mother tongue.

Ce travail de diplôme, fruit de trois années d'études n'aurait jamais pu se réaliser sans l'aide de nombreuses personnes. En essayant de n'oublier personne, et — afin de ne pas risquer de froisser ceux qui n'auraient pas été oubliés — en voici une liste chronologique que j'espère exhaustive.

Tout a commencé par le soutien incondtionnel durant toute la phase finale de mes études de mon amie Noémie.

Ensuite, et plus professionnellement, j'aimerais remercier chaleureusement Omar ABOU KHALED qui a eu la gentillesse de me donner l'opportunité d'exécuter mon travail de diplôme *ex cathedra*. Il me faut également remercier Jean-Yves LE MEUR pour m'avoir non seulement accueilli au sein du CERN, mais également pour tous les conseils qu'il a su me prodiguer. Aider au développement d'un logiciel de l'envergure de CDSware ne peut se faire aisément ; il me faut par conséquent remercier l'architecte en chef de ce projet, Tibor ŠIMKO qui a passé un nombre incalculable d'heures à discuter de problèmes techniques, et à éclairer ma route ! Il a également contribué à la relecture de ce document.

Il me faut également remercier mes collègues qui ont également répondu à mes nombreuses questions, spécialement Paulo CABRAL, dans les deux premières semaines, puis Martin VESELY et Alberto PEPE (également pour la traduction italienne du résumé) qui ont été d'une disponibilité sans faille.

Enfin, pour ses conseils sur l'élaboration du présent document et son suivi, je tiens également à remercier vivement Houda CHABBI DRISSI.

Bibliography

- [1] *About cern*, CERN,
<http://public.cern.ch/public>.
- [2] *Apache http server project*, The Apache Software Foundation,
<http://httpd.apache.org>.
- [3] *Autoconf - gnu project*, Free Software Foundation,
<http://clisp.cons.org>.
- [4] *Clisp - an ansi common lisp implementation*, CLISP,
<http://clisp.cons.org>.
- [5] *Html 4.01 specification*, World Wide Web Consortium,
<http://www.w3.org/TR/REC-html40>.
- [6] *Knowledge*, Wikipedia, the free encyclopedia,
<http://en.wikipedia.org/wiki/Knowledge>.
- [7] *Mod_python - apache/python integration*, The Apache Software Foundation,
<http://www.modpython.org>.
- [8] *Mysql for python*, MySQLdb,
<http://www.sourceforge.net/projects/mysql-python>.
- [9] *Mysql: The world's most popular open source database*, MySQL AB,
<http://www.modpython.org>.
- [10] *Php: Hypertext preprocessor*, The PHP Group,
<http://www.php.net>.
- [11] *Protocol for metadata harvesting*, Open Archive initiative,
<http://www.openarchives.org.org>.
- [12] *Python programming language*, Python Software Foundation,
<http://www.python.org>.
- [13] *Server databases clash*, eWeek,
<http://www.eweek.com/article2/0,4149,293,00.asp>.
- [14] *Website meta language (wml)*, WML,
<http://www.thewml.org>.

- [15] *World wide web - summary*, W3C: World Wide Web Consortium,
<http://www.w3.org/summary>.
- [16] Paulo Cabral, *Personalization of Modern Digital Library Systems: Extending CDSware to a social software*, Master's thesis, EPFL, 2005.
- [17] Alberto Pepe, Thomas Baron, Maja Gracco, Jean-Yves Le Meur, Nicholas Robinson, Tibor Šimko, and Martin Vesely, *Cern document server software: the integrated digital library*, From Author to Reader : Challenges for the Digital Content Chain. 9th ICCC International Conference on Electronic Publishing (10 Jun 2005).
- [18] Tibor Šimko, *Vademecum and essays [restricted access]*, CERN,
<http://simko.home.cern.ch/simko/vademecum/index.html>.

Glossary

CDS	CERN Document Server, the repository for publications in CERN	7
CDSware	CERN Document Server Software	1
CDSweb	The release of CDSware installed at CERN. One of the world's biggest digital library in domain of particle physics	7
CLI	Command Line Interface.	21
CVS	The Concurrent Versions System (CVS), also known as the Concurrent Versioning System, implements a version control system: it keeps track of all work and all changes in a set of files, typically the implementation of a software project, and allows several (potentially widely separated) developers to collaborate. CVS has become popular in the open-source world. CVS is released under the GNU General Public License.	31
DBMS	Database Management System; a DBMS is a program (or more typically, a suite of them) designed to manage a database (a large set of structured data), and run operations on the data requested by numerous clients	11
Dublin Core	Dublin Core is a metadata standard for describing digital objects (including webpages) to enhance visibility, accessibility and interoperability. It is used as exchange format by OAi-PMH	11
HCI	Human-Computer Interface, the technical means by which a human interacts with hardware or software.	34
LOC	Library Of Congress; official depository of United States publications	11

MARC 21	MAchine Readable Cataloging record; edict of the Library of Congress (LOC), MARC 21 is a standard format for the representation of bibliographic data	11
Metadata	literally “data about data”, is information that describes another set of data. A common example is a library catalog card, which contains data about the contents and location of a book: It is data about the data in the book referred to by the card. Other common contents of metadata include the source or author of the described dataset, how it should be accessed, and its limitations.	2
OAI-PMH	Open Archive initiative Metadata Harvesting Protocol. A protocol which defines ways of exchanging meta-data between distinct digital libraries	2

Appendix A

Initial specifications of project

Diploma Project

Extending CDSware with social tools

Specifications

GREGORY FAVRE, EIF - gregory.favre@cern.ch

November 14, 2005

1 Project overview

1.1 CDSware

The CERN Document Server Software (CDSware) is the software developed by, maintained by, and used at, the CERN Document Server.

At CERN, CDSware manages over 500 collections of data, consisting of over 800,000 bibliographic records, including 300,000 fulltext documents. These collections include preprints, articles, books, journals, photographs, and more.

CDSware is licensed under GPL and is thus free software. This software is currently being used by a dozen institutions around the globe.

1.2 Social software?

Although CDSware is a very complete system containing many extended features, the current personalization features are under-employed. Only 2% of the users register themselves and access these tools. However, they seem convinced by these features, as 60% of the registered users use the current collaborative tools (search baskets, alerts) already put in place.

The main goal of this diploma project is to extend and enhance the collaborative features offered by CDSware and allow users to collaborate in more effective ways.

2 Work

An analysis of user needs was already carried out by a student¹. As a result of this, partial implementation and design were carried out. The goal is to continue his work, by implementing the core modules and refining the design.

¹Paulo Cabral - SIN - EPFL

The entity-relationship diagram of the collaborative part of the database, received at the beginning of the work can be found below [fig. 1]. This diagram needs to be refined, as it doesn't reflect every aspect of the interaction of the syntax incorrectness.

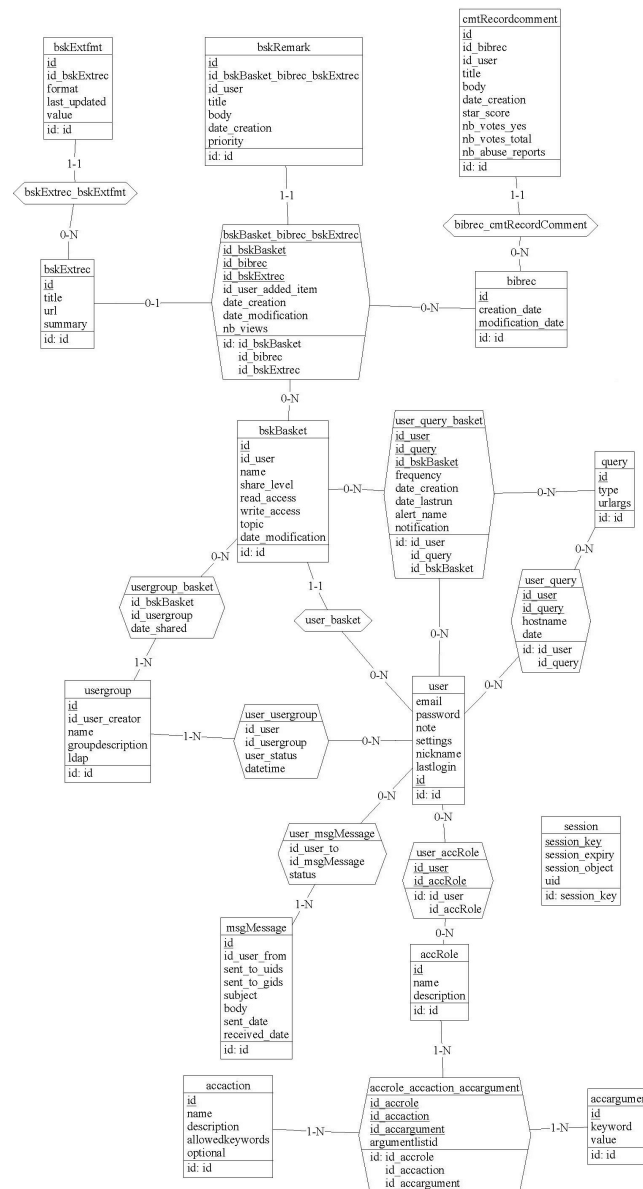


Figure 1: Initial stage of the database modeling, as received. Note that this diagram has to be refined

2.1 WebMessage

In CDSWare, due to policy rules, the email address of a user is never given to another. The side-effect of this policy is that, in order to use it collaboratively, users need an intern messaging system. This feature should look like any webmail system.

Implementation will be done, with particular care in integration with other modules.

2.2 WebBasket

Users can save the bibliographic info of interesting documents in a basket. A basket should be either public, private or restricted for a group. Organization of private baskets by topics should be made available. A complete commenting system for entries in baskets should take place. A system of access privileges (read, read comments, add comment, add article, etc.) must be developed.

A module already exists but it will need to be adapted/rewritten in order to make it possible to use it collaboratively. Export and import functions are currently developped by a CERN fellow and must be integrated in the new webbasket module. Design will be completed and the full implementation is required.

2.3 WebSession

If possible with time, this has to be refined for the use of groups. Design is partially done.

2.4 Summary

Functionality	Current	Forthcoming
User managing	Users can create accounts. Each user who has an account can have personal baskets.	Users can be organized in user groups. User groups can possess private baskets.
Information sharing	Users can make comments on articles. They possess baskets which can be public.	Users can communicate through a messaging system, or through commenting at basket level.
Information access	Baskets are either public or private	Baskets can be shared at user level or group level. Special rights are applied at basket level (read comments, write comments, add articles, etc.).
Information organization	Users can create several baskets	Users can create topics which contain private baskets. They can access baskets at group level (possibly also organized by topic). They can import other users' baskets.
Information source	Users can add CDS articles to their baskets	Users can add CDS or external entries. External entries can be added by giving a name and a URL, or by providing a BibTeX file. Baskets can be exported in BibTeX.

3 Organization

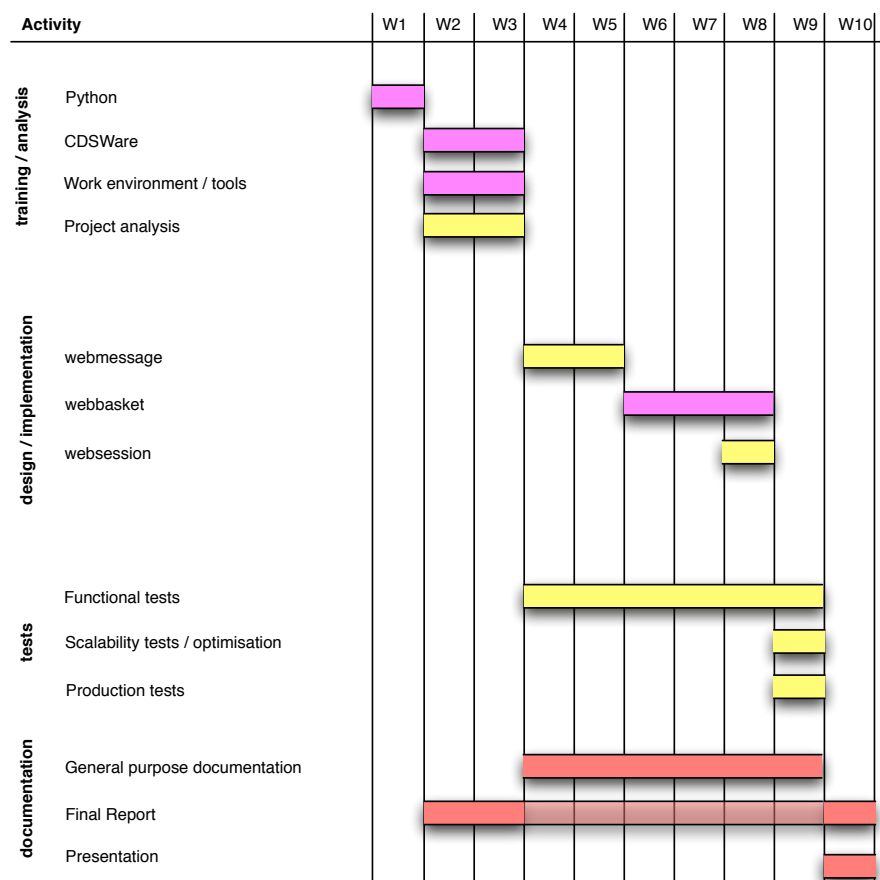
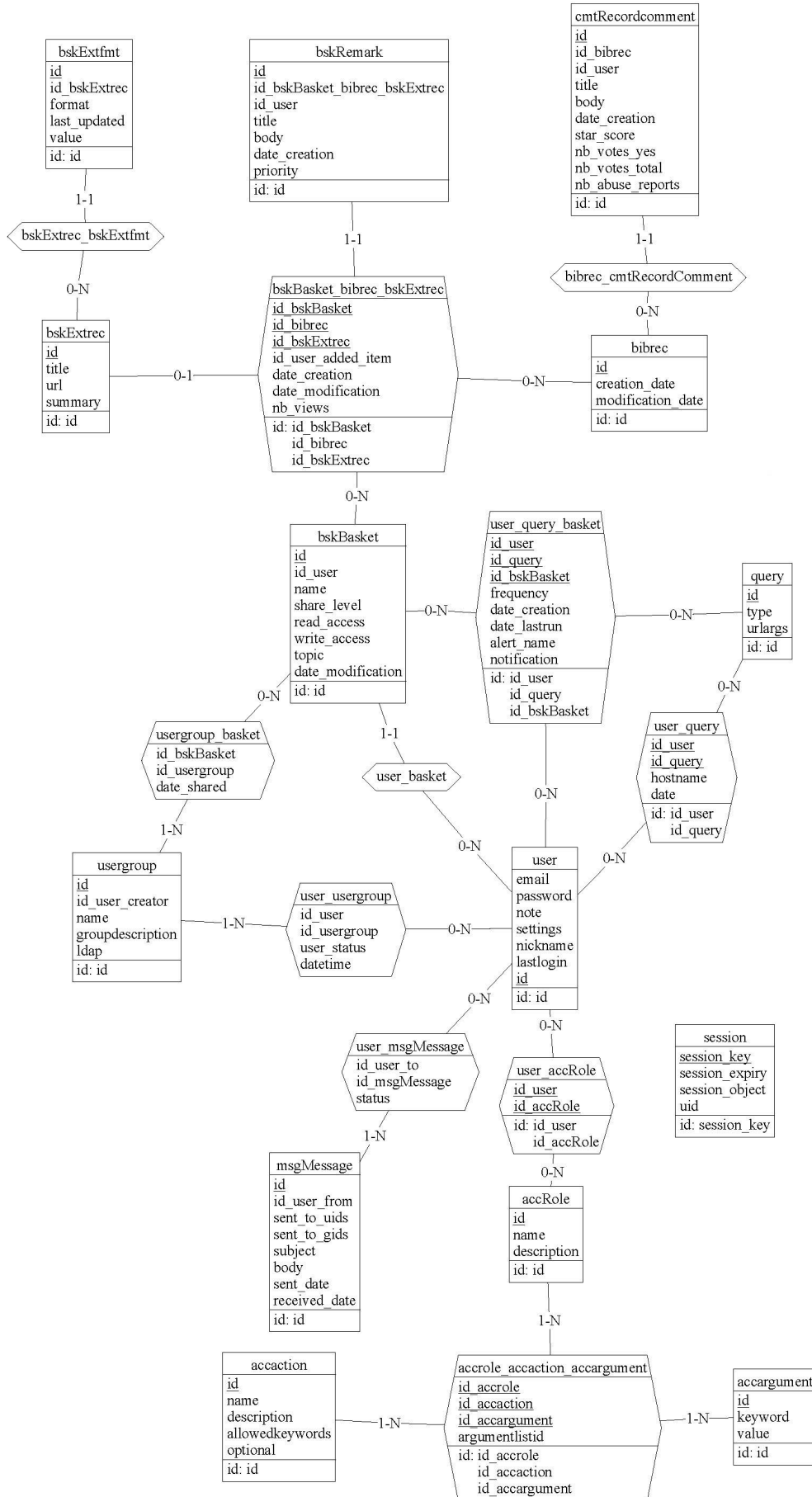


Figure 2: Planing

Appendix B

Paulo Cabral's prototypes

B.1 Database



B.2 WebMessage

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

[Home](#) > [Your Account](#) > [Your Messages](#)

[Message Inbox](#) | [Write a message](#)

Inbox: 2 new messages

Quota at 65%

	Subject	Sender	Date	Delete
New	Looking for Paulo	Pepe	24 Sept. 2005 at 14:43	<input type="checkbox"/>
New	Invitation to join a group	Voss	24 Sept. 2005 at 13:22	<input type="checkbox"/>
	Invitationa to join a group	Linssen	13 Sept. 2005 at 19:04	<input type="checkbox"/>
	Interesting item	Navarro	10 Sept. 2005 at 11:33	<input type="checkbox"/>

DELETE

Figure B.1: WebMessage: inbox



Figure B.2: WebMessage: Write a message

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

SearchSubmitPersonalizeHelp

[Home](#) > [Your Account](#) > Your Messages

Write a Message

[Message Inbox](#) | [Write a message](#)

[Find a user](#) | [Find a group](#)

Send a message
To: Users
Groups
Subject:
Message:

Tip: You can create a *reminder* by sending the message at a later date.

Date: / /

SEND LATER

Figure B.3: WebMessage: write a reminder

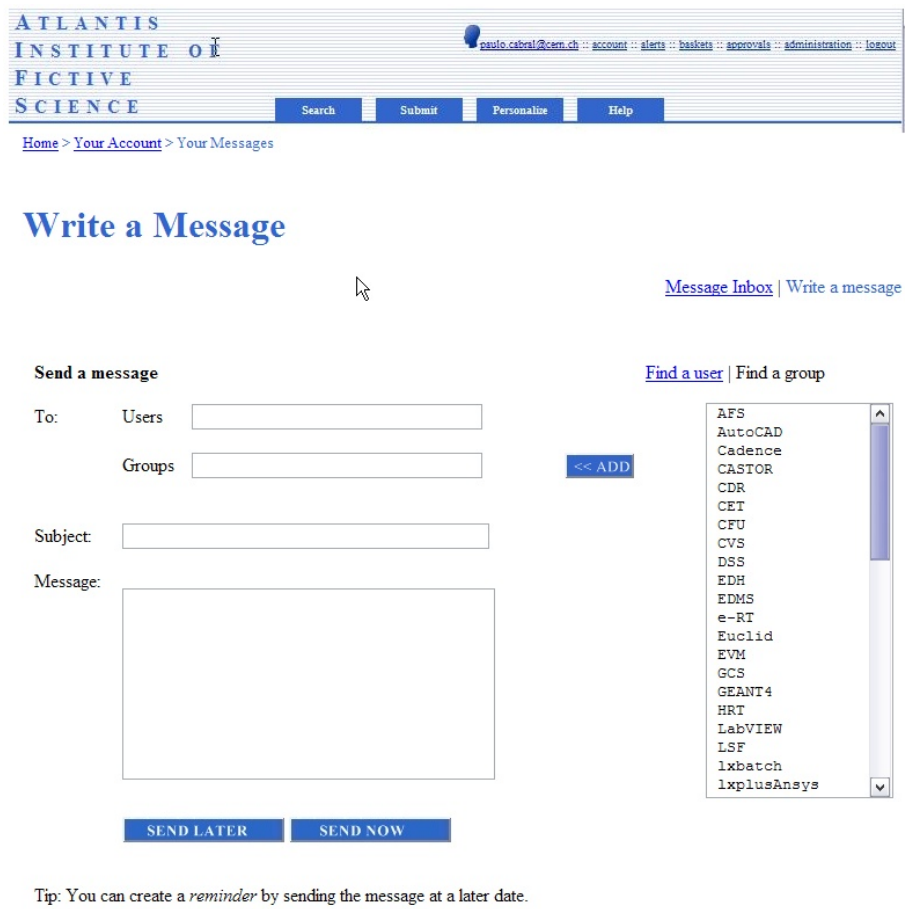



Figure B.4: WebMessage: find a group

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

 paulo.cabral@cem.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

[Search](#) [Submit](#) [Personalize](#) [Help](#)

[Home](#) > [Your Account](#) > Your Messages

Write a Message

[Message Inbox](#) | [Write a message](#)

Send a message

To:

Users

Groups

Subject:

Message:

SEND LATER

SEND NOW

Find a user | [Find a group](#)

<< ADD

Alvarez

Blaising

Burckhart

Burri

Correia

Dittus

Ellis

Elsing

Farthouat

Fassnacht

Fraile

Froidevaux

Grafstorm

Hatch

Henriques

Hervas

Kate

Kotamaki

Linszen

Mapelli

Mclaren

Tip: You can create a *reminder* by sending the message at a later date.

Figure B.5: WebMessage: find a user

B.3 WebBasket



The screenshot shows the WebBasket interface. At the top, the header reads "ATLANTIS INSTITUTE OF FICTIVE SCIENCE". To the right, a user profile "paulo.cabral@cern.ch" is logged in, with links for "account", "alerts", "baskets", "approvals", "administration", and "logout". Below the header is a navigation bar with buttons for "Search", "Submit", "Personalize", and "Help". A breadcrumb trail reads "Home > Your Account > Your Baskets > CERN Experiments > ALEPH".

Your Baskets

You are viewing your ALEPH basket.

Select a different basket:

There isn't any alert related to this basket.

 **ALEPH (Public)**


 In the search of ...	(2 comments)
 Detection of ...	(10 comments)
 Theory of ...	(3 comments)
 ALEPH	(No comments)
 Cornell Experiment	(2 comments)

Figure B.6: WebBasket: View a basket

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE


paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalise Help

Home > Your Account > Your Baskets > CERN Experiments > ALEPH

View BasketItem

Added by user *Selgio* on 14 September 2005 at 12:01 2 Comments | 81 Visits

 **Cornell Experiment**
URL: <http://cuaes.cornell.edu/CUAESWeb/home.htm>
Description: The Accessible Webcast Seminars (AWSem) project at Cornell is aimed at making participation in selected seminars possible from anywhere in the world using commonly available hardware and software.

Tell a friend or group about this item:

Select nickname: or select group:

Enter message:

Hi, I came accross this basket item and thought you may be interested
<http://pcuds11.cern.ch/yourbaskets.py/item=43>

SEND MESSAGE

Set an alert for this basket item:

Send you a notification when new comments are added to this basket item via ☒ Email ☐ Message

SET ALERT

There are 2 comments abouts this item:

Voss writes on 15 September 2005 at 16:32 [Quote](#)
This could be useful.
Quoting the website:
"A database of in-progress and planned remote operations and experiments activities will soon be available online. Please send information on remote operations and experiments activities at your lab to Fulvia Pilat . Include if possible: Host lab name, experiment name, time scale for activity, description of activity and objectives, collaborating partners, benefits to be realized, factors which limit full remote control, special requirements for new equipment/software, controls in use, contact names, and status."
[Quote](#)

Dittus writes on 15 September 2005 at 16:32 [Quote](#)
Nice!
WRITE A COMMENT

Figure B.7: WebBasket View an item

ATLANTIS
INSTITUTE OF
FICTIONAL
SCIENCE

[paulo.cabral@cern.ch](#) :: [account](#) :: [alerts](#) :: [baskets](#) :: [approvals](#) :: [administration](#) :: [logout](#)


[Search](#) [Submit](#) [Personalize](#) [Help](#)

[Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > ALEPH

View BasketItem

Added by user *Selgio* on 14 September 2005 at 12:01

2 Comments | 81 Visits



Cornell Experiment

URL: <http://cuaes.cornell.edu/CUAESWeb/home.htm>

Description: The Accessible Webcast Seminars (AWSem) project at Cornell is aimed at making participation in selected seminars possible from anywhere in the world using commonly available hardware and software.

Tell a friend or group about this item:

Select nickname: or select group:

Enter message:

Hi, I came across this basket item and thought you may be interested
<http://pcuds11.cern.ch/yourbaskets.py/item=43>

[SEND MESSAGE](#)

Set an alert for this basket item:

Send you a notification when new comments are added to this basket item via ☒ Email ☐ Message

[SET ALERT](#)

Figure B.8: WebBasket: View an item

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalise Help

[Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > [ALEPH](#)

Edit Baskets

You are viewing your ALEPH basket from your CERN Experiments Category.

Select a different basket:

Rename this basket:

Legend:
 Delete item
 Copy to another category
 Move down in basket
 Move up in basket

ALEPH (Public)

In the search of ...	(2 comments)
Detection of ...	(10 comments)
Theory of ...	(3 comments)
ALEPH	(No comments)
Cornell Experiment	(2 comments)

Sort by or

Figure B.9: WebBasket: Edit a basket

The screenshot shows a web interface for 'ATLANTIS INSTITUTE OF FICTIONAL SCIENCE'. The header includes a logo on the left and a user profile 'paulo.cabral@cern.ch' with links to 'account', 'alerts', 'baskets', 'approvals', 'administration', and 'logout' on the right. Below the header is a navigation bar with buttons for 'Search', 'Submit', 'Personalise', and 'Help'. A breadcrumb trail reads 'Home > Your Account > Your Baskets > CERN Experiments > ALEPH'. The main heading is 'Add to Baskets'. Below this, it says 'Adding to your ALEPH basket' followed by a red arrow pointing to 'Step 1: Type of basket item' and 'Step 2: Basket item'. A mouse cursor points to a blue-bordered box titled 'Step 1 of 2' and 'Type of Basket Item'. Inside this box, the question 'What would you like to add to this basket?' is followed by two radio button options: 'Internal CDS Record (Articles, Books, ...)' (which is selected) and 'External Record (Bookmark, External document, Note)'. At the bottom right of the box are 'Cancel' and 'Next →' buttons.

ATLANTIS
INSTITUTE OF
FICTIONAL
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalise Help

Home > Your Account > Your Baskets > CERN Experiments > ALEPH

Add to Baskets

Adding to your ALEPH basket
-> Step 1: Type of basket item
Step 2: Basket item

Step 1 of 2

Type of Basket Item

What would you like to add to this basket?

☒ Internal CDS Record (Articles, Books, ...)

☐ External Record (Bookmark, External document, Note)

Cancel Next →

Figure B.10: WebBasket: add an item step 1

The screenshot shows the Atlantis Institute of Fictive Science website. The header includes the logo and navigation links: [paslo.cebral@cern.ch](#), [account](#), [alerts](#), [baskets](#), [approvals](#), [administration](#), and [logout](#). Below the header are buttons for [Search](#), [Submit](#), [Personalize](#), and [Help](#). The breadcrumb trail reads: [Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > [ALEPH](#).

Add to Baskets

Adding to your ALEPH basket

- ✓ Step 1: Type of basket item
- > Step 2: Basket item

Step 2 of 2

Basket Item

Title

URL

Description

Add additional bibliographic detail to this record?

☒ No

☐ Yes

[← Back](#) [Cancel](#) [Finish](#)

Figure B.11: WebBasket: Add an item, step 2

ATLANTIS
INSTITUTE OF
FICTIONAL
SCIENCE

[paulo.cabral@cern.ch](#) :: [account](#) :: [alerts](#) :: [baskets](#) :: [approvals](#) :: [administration](#) :: [logout](#)

[Search](#) [Submit](#) [Personalise](#) [Help](#)

[Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > ALEPH

Share Basket

You are changing the sharing of your ALEPH basket from your Personal category.

Select a different basket: [SELECT](#)

☒ ALEPH (Public)

Sharing of this basket is set to: ☒ public

Read access: Everyone

Write access: Everyone

[CHANGE SHARING](#)

Copy share settings from basket: [COPY SETTINGS](#)

Figure B.12: WebBasket: Share a basket, step 1

The screenshot shows the Atlantis Institute of Fictive Science WebBasket interface. The header includes the logo and navigation links: paulo.cabral@cern.ch, account, alerts, baskets, approvals, administration, and logout. Below the header is a breadcrumb trail: Home > Your Account > Your Baskets > CERN Experiments > ALEPH. The main heading is 'Change Sharing'. Below it, instructions state: 'Change the share settings of your ALEPH basket.' followed by '-> Step 1: Type of sharing' and 'Step 2: Access privileges'. A mouse cursor points to the right. The main content area is titled 'Step 1 of 2' and 'Type of sharing'. It asks 'With whom would you like to share this basket?' and provides three radio button options: 'No one (private basket)' (selected), 'Group (restricted basket)', and 'Everyone (public basket)'. At the bottom right are 'Cancel' and 'Next ->' buttons.

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalize Help

Home > Your Account > Your Baskets > CERN Experiments > ALEPH

Change Sharing

Change the share settings of your ALEPH basket.

-> Step 1: Type of sharing
Step 2: Access privileges

Step 1 of 2

Type of sharing

With whom would you like to share this basket?

☒ No one (private basket)

☐ Group (restricted basket)

☐ Everyone (public basket)

Cancel Next ->

Figure B.13: WebBasket: Share a basket, step 2

The screenshot shows a web application interface for 'ATLANTIS INSTITUTE OF FICTIVE SCIENCE'. The header includes a user profile 'paulo.cabral@cern.ch' and navigation links: 'account', 'alerts', 'baskets', 'approvals', 'administration', and 'logout'. Below the header is a breadcrumb trail: 'Home > Your Account > Your Baskets > CERN Experiments > ALEPH'. The main heading is 'Change Sharing'. Below it, instructions state: 'Change the share settings of your ALEPH basket.' followed by a progress indicator: '✓ Step 1: Type of sharing (private)' and '-> Step 2: Access privileges'. The current step is 'Step 2 of 2' with the title 'Read/Write Access'. It contains two settings: 'Read access: Only you' and 'Write access: Only you'. At the bottom right are three buttons: '← Back', 'Cancel', and 'Finish'.

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalise Help

[Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > [ALEPH](#)

Change Sharing

Change the share settings of your ALEPH basket.

✓ Step 1: Type of sharing (private)
-> Step 2: Access privileges

Step 2 of 2

Read/Write Access

Read access: Only you

Write access: Only you

← Back Cancel Finish

Figure B.14: WebBasket: Share a basket, step 3

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

paulo.csbral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalize Help

[Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > [ALEPH](#)

Change Sharing

Change the share settings of your ALEPH basket.

- ✓ Step 1: Type of sharing (public)
- Step 2: Access privileges
 - > a) Group selection
 - b) Read/write access

Step 2 of 2

Group Selection

☒ Select an existing group

☐ Create a new group

Group name:

Group description:

(You will have to invite users to join your group later).

Figure B.15: WebBasket: Share a basket, step 4

The screenshot shows the Atlantis Institute of Fictive Science website. The header includes the logo and navigation links: Home, Your Account, Your Baskets, CERN Experiments, ALEPH, Search, Submit, Personalise, and Help. The user is logged in as paulo.cabral@cern.ch. The main content area is titled 'Change Sharing' and instructs the user to change the share settings of their ALEPH basket. It shows two steps: Step 1 (Type of sharing (public)) and Step 2 (Access privileges). Step 2 is further divided into 'a) Group selection' and 'b) Read/write access'. The current step is 'Group Selection', which is highlighted with a blue border. It shows a list of groups that ALEPH consists of: Schlatter (moderator), Alvarez, Blaising, Linssen, Salicio, Prieto, Fraile, Rembser, and Taureg. Navigation buttons for Back, Cancel, and Next are at the bottom.

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

Home > Your Account > Your Baskets > CERN Experiments > ALEPH

Change Sharing

Change the share settings of your ALEPH basket.

- ✓ Step 1: Type of sharing (public)
- Step 2: Access privileges
 - > a) Group selection
 - b) Read/write access

Step 2 of 2

Group Selection

Group ALEPH consists of:

- Schlatter (moderator)
- Alvarez
- Blaising
- Linssen
- Salicio
- Prieto
- Fraile
- Rembser
- Taureg

← Back Cancel Next →

Figure B.16: WebBasket: Share a basket, step 5

The screenshot shows the WebBasket interface. At the top, the header reads 'ATLANTIS INSTITUTE OF FICTIONAL SCIENCE'. To the right, a user profile 'paulo.cabral@cern.ch' is shown with links for 'account', 'alerts', 'baskets', 'approvals', 'administration', and 'logout'. Below the header is a navigation bar with buttons for 'Search', 'Submit', 'Personalize', and 'Help'. A breadcrumb trail reads 'Home > Your Account > Your Baskets > CERN Experiments > ALEPH'. The main heading is 'Change Sharing'. Below it, a message says 'Change the share settings of your ALEPH basket.' followed by a progress list: '✓ Step 1: Type of sharing (public)', 'Step 2: Access privileges', '✓ a) Group selection (ALEPH)', and '-> b) Read/write access'. The current step is 'Step 2 of 2' and is titled 'Read/Write Access'. It contains two sections: 'Read access: Who should be able to see your basket and its contents?' with radio buttons for 'Group' (selected) and 'Everyone'; and 'Write access: Should the ALEPH group members be allowed to comment and add basket items to your shared basket?' with radio buttons for 'Yes' (selected) and 'No'. At the bottom are three buttons: '← Back', 'Cancel', and 'Next →'.

ATLANTIS
INSTITUTE OF
FICTIONAL
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalize Help

Home > Your Account > Your Baskets > CERN Experiments > ALEPH

Change Sharing

Change the share settings of your ALEPH basket.

- ✓ Step 1: Type of sharing (public)
- Step 2: Access privileges
 - ✓ a) Group selection (ALEPH)
 - > b) Read/write access

Step 2 of 2

Read/Write Access

Read access: Who should be able to see your basket and its contents?

☒ Group
☐ Everyone

Write access: Should the ALEPH group members be allowed to comment and add basket items to your shared basket?

☒ Yes
☐ No

← Back Cancel Next →

Figure B.17: WebBasket: Share a basket, step 6

The screenshot displays a web interface for the Atlantis Institute of Fictive Science. The header includes the site name, a user profile for paulo.cabral@cern.ch, and navigation links. A breadcrumb trail shows the path from Home to ALEPH. The main heading is 'Change Sharing', followed by instructions to change share settings for the ALEPH basket. A progress list shows Step 1 as complete and Step 2 as the current step. Step 2 is titled 'Finish' and instructs the user to invite specific users. At the bottom of the step box are 'Back', 'Cancel', and 'Finish' buttons.

ATLANTIS
INSTITUTE OF
FICTIVE
SCIENCE

paulo.cabral@cern.ch :: account :: alerts :: baskets :: approvals :: administration :: logout

Search Submit Personalize Help

[Home](#) > [Your Account](#) > [Your Baskets](#) > [CERN Experiments](#) > [ALEPH](#)

Change Sharing

Change the share settings of your ALEPH basket.

- ✓ Step 1: Type of sharing (public)
- ✓ Step 2: Access privileges
 - a) Group selection (ALEPH)
 - b) Read/write access

Step 2 of 2

Finish

Please invite specific users to join your newly created group on the next page

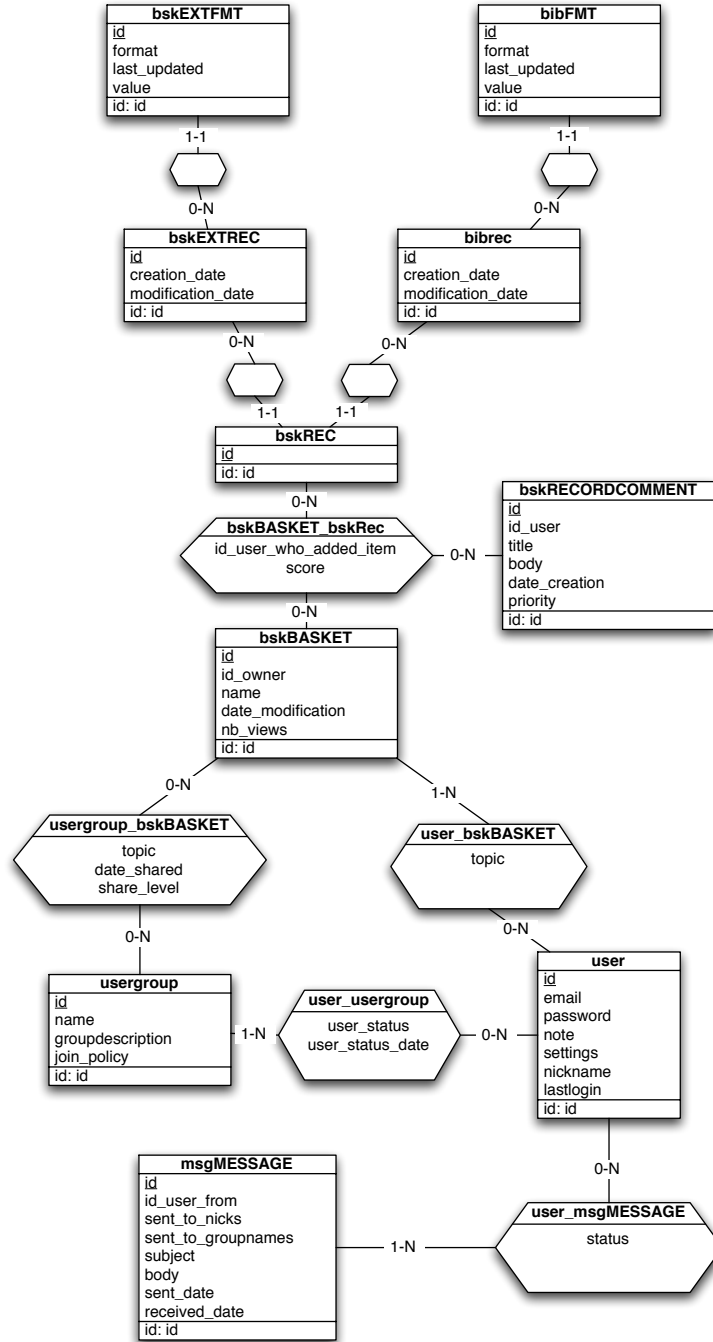
[← Back](#) [Cancel](#) [Finish](#)

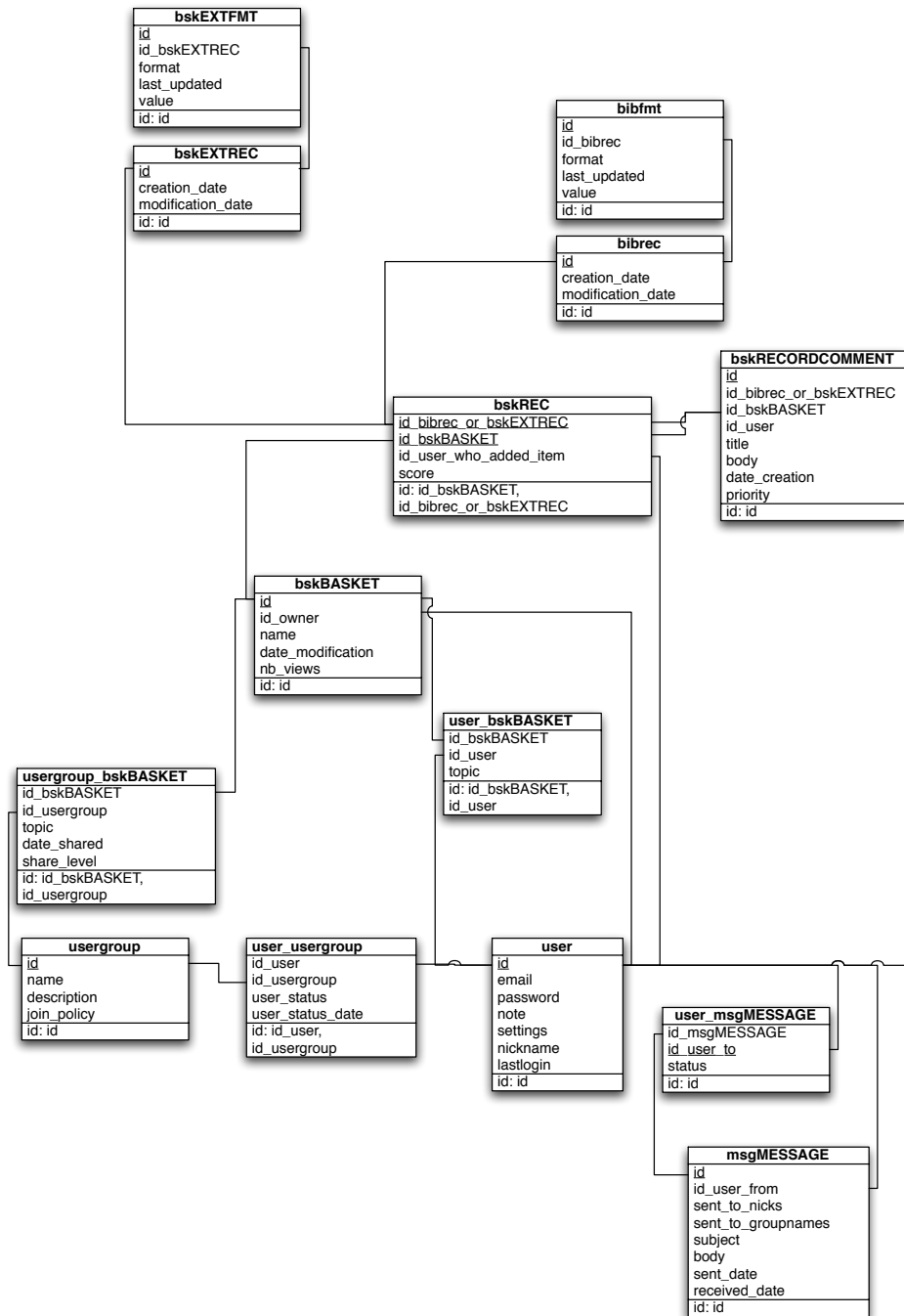
Figure B.18: WebBasket: Share a basket, step 7

Appendix C

Complete database diagrams

On the next two pages, you'll find the complete ER and Relational diagrams.





Appendix D

API

D.1 WebMessage

Help on module webmessage:

NAME

webmessage - webmessage module, messaging system

FILE

/home/cdsware/lib/python/cdsware/webmessage.py

FUNCTIONS

```
account_new_mail(uid, ln='en')
    display new mail info for myaccount.py page.
    @param uid: user id (int)
    @param ln: language
    @return html body
```

```
escape_string(...)
    escape_string(s) -- quote any SQL-interpreted characters in string s.
```

Use connection.escape_string(s), if you use it at all.
_mysql.escape_string(s) cannot handle character sets. You are probably better off using connection.escape(o) instead, since it will escape entire sequences as well as strings.

```
get_navtrail(ln='en', title='')
    gets the navtrail for title...
    @param title: title of the page
    @param ln: language
    @return HTML output
```

```
localtime(...)
    localtime([seconds]) -> (tm_year,tm_mon,tm_day,tm_hour,tm_min,tm_sec,tm_wday,tm_yday,tm_isdst)
```

Convert seconds since the Epoch to a time tuple expressing local time.
When 'seconds' is not passed in, convert the current time instead.

```
mktime(...)
    mktime(tuple) -> floating point number
```

Convert a time tuple in local time to seconds since the Epoch.

```
perform_request_delete_all(uid, confirmed=0, ln='en')
    Delete every message for a given user
    @param uid: user id (int)
    @param confirmed: 0 will produce a confirmation message
    @param ln: language
    @return a (body, errors, warnings) tuple
```

```
perform_request_delete_msg(uid, msgid, ln='en')
    Delete a given message from user inbox
    @param uid: user id (int)
    @param msgid: message id (int)
    @param ln: language
    @return a (body, errors, warning tuple)
```

```
perform_request_display(uid, errors=[], warnings=[], infos=[], ln='en')
    Displays the user's Inbox
    @param uid: user id

    @return a (body, [errors], [warnings]) formed tuple
```

```
perform_request_display_msg(uid, msgid, ln='en')
    Displays a specific message
    @param uid: user id
    @param msgid: message id

    @return a (body, errors[], warnings[]) formed tuple
```

```
perform_request_send(uid, msg_to_user='', msg_to_group='', msg_subject='', msg_body='')
    send a message. if unable return warnings to write page
    @param uid: id of user from (int)
    @param msg_to_user: comma separated usernames (recipients) (str)
    @param msg_to_group: comma separated groupnames (recipients) (str)
    @param msg_subject: subject of message (str)
    @param msg_body: body of message (str)
    @param msg_send_year: send this message on year x (int)
    @param msg_send_month: send this message on month y (int)
    @param msg_send_day: send this message on day z (int)
    @param ln: language
    @return a (body, errors, warnings) tuple
```

```
perform_request_write(uid, msg_reply_id='', msg_to='', msg_to_group='', ln='en')
```

```

    Display a write a message page.
    @param uid: user id (int)
    @param msg_reply_id: if this message is a reply to another, other's ID (int)
    @param msg_to: comma separated usernames (string)
    @param msg_to_group: comma separated groupnames (string)
    @param ln: language
    @return a (body, errors, warnings) tuple

perform_request_write_with_search(msg_to_user='', msg_to_group='', msg_subject='', msg_body=
    Display a write message page, with prefilled values
    @param msg_to_user: comma separated usernames (str)
    @param msg_to_group: comma separated groupnames (str)
    @param msg_subject: message subject (str)
    @param msg_bidy: message body (string)
    @param msg_send_year: year to send this message on (int)
    @param msg_send_month: month to send this message on (int)
    @param msg_send_day: day to send this message on (int)
    @param users_to_add: list of usernames ['str'] to add to msg_to_user
    @param groups_to_add: list of groupnames ['str'] to add to msg_to_group
    @param user_search_pattern: will search users with this pattern (str)
    @param group_search_pattern: will search groups with this pattern (str)
    @param mode_user: if 1 display user search box, else group search box
    @param add_values: if 1 users_to_add will be added to msg_to_user field..
    @param ln: language
    @return a (body, errors, warnings) formed tuple.

```

DATA

```

__lastupdated__ = '$Date: 2005/11/16 15:09:23 $'
cdslang = 'en'
cfg_webmessage_days_before_delete_orphans = 60
cfg_webmessage_error_messages = {'ERR_WEBMESSAGE_NOMESSAGE': " This me...
cfg_webmessage_max_nb_of_messages = 30
cfg_webmessage_max_size_of_message = 20000
cfg_webmessage_roles_without_quota = ['superadmin']
cfg_webmessage_separator = ','
cfg_webmessage_status_code = {'NEW': 'N', 'READ': 'R', 'REMINDER': 'M'...
datetext_default = '0000-00-00 00:00:00'
webmessage_templates = <cdsware.webmessage_templates.Template instance...

```

D.2 DateUtils

Help on module dateutils:

NAME

dateutils

FILE

/home/cdsware/lib/python/cdsware/dateutils.py

DESCRIPTION

API for date conversion and date related GUI creation.

Lexicon

datetext:

textual format => 'YEAR-MONTH-DAY HOUR:MINUTE:SECOND'

e.g. '2005-11-16 15:11:44'

default value: '0000-00-00 00:00:00'

datestruct:

tuple format => see <http://docs.python.org/lib/module-time.html>

(YEAR, MONTH, DAY, HOUR, MINUTE, SECOND, WEEKDAY, YEARDAY, DAYLIGHT)

e.g. (2005, 11, 16, 15, 11, 44, 2, 320, 0)

default value: (0, 0, 0, 0, 0, 0, 0, 0, 0)

dategui:

textual format for output => 'DAY MONTH YEAR, HOUR:MINUTE'

e.g. '16 nov 2005, 15:11'

default value: _("N/A")

FUNCTIONS

convert_datestruct_to_dategui(datestruct, ln='en')

Convert:

(2005, 11, 16, 15, 11, 44, 2, 320, 0) => '16 nov 2005, 15:11'

Month is internationalized

convert_datestruct_to_datetext(datestruct)

Convert:

(2005, 11, 16, 15, 11, 44, 2, 320, 0) => '2005-11-16 15:11:57'

convert_datetext_to_dategui(datetext, ln='en')

Convert:

'2005-11-16 15:11:57' => '16 nov 2005, 15:11'

Month is internationalized

convert_datetext_to_datestruct(datetext)

Convert:

'2005-11-16 15:11:57' => (2005, 11, 16, 15, 11, 44, 2, 320, 0)

create_day_selectbox(name, selected_day=0, ln='en')

Creates an HTML menu for day selection. (0..31 values).

@param name: name of the control (i.e. name of the var you'll get)

@param selected_day: preselect a day. Use 0 for the label 'Day'

@param ln: language of the menu

@return html a string

create_month_selectbox(name, selected_month=0, ln='en')

Creates an HTML menu for month selection. Value of selected field is numeric


```

    @param name: name of the control (your form will be sent with name=value...)
    @param selected_month: preselect a month. use 0 for the Label 'Month'
    @param ln: language of the menu
    @return html as string

create_year_inputbox(name, value=0)
    Creates an HTML field (simple input) for year selection.
    @param name: name of the control (i.e. name of the variable you'll get)
    @param value: prefilled value (int)
    @return html as string

create_year_selectbox(name, from_year=-1, length=10, selected_year=0, ln='en')
    Creates an HTML menu (dropdownbox) for year selection.
    @param name: name of control( i.e. name of the variable you'll get)
    @param from_year: year on which to begin. if <0 assume it is current year
    @param length: number of items in menu
    @param selected_year: initial selected year (if in range), else: label is selected
    @param ln: language
    @return html as string

get_datestruct(year, month, day)
    year=2005, month=11, day=16 => (2005, 11, 16, 0, 0, 0, 2, 320, -1)

get_datetext(year, month, day)
    year=2005, month=11, day=16 => '2005-11-16 00:00:00'

get_i18n_day_name(day_nb, display='short', ln='en')
    get the string representation of a weekday, internationalized
    @param day_nb: number of weekday UNIX like.
        => 0=Sunday
    @param ln: language for output
    @return the string representation of the day

get_i18n_month_name(month_nb, display='short', ln='en')
    get a non-numeric representation of a month, internationalized.
    @param month_nb: number of month, (1 based!)
        => 1=jan,...,12=dec
    @param ln: language for output
    @return the string representation of month

localtime(...)
    localtime([seconds]) -> (tm_year,tm_mon,tm_day,tm_hour,tm_min,tm_sec,tm_wday,tm_yday,tm_isdst)

    Convert seconds since the Epoch to a time tuple expressing local time.
    When 'seconds' is not passed in, convert the current time instead.

strftime(...)
    strftime(format[, tuple]) -> string

    Convert a time tuple to a string according to a format specification.

```

See the library reference manual for formatting codes. When the time tuple is not present, current time as returned by `localtime()` is used.

```
strptime(...)
    strptime(string, format) -> struct_time
```

Parse a string to a time tuple according to a format specification.
See the library reference manual for formatting codes (same as `strftime()`).

DATA

```
cdslang = 'en'
datestruct_default = (0, 0, 0, 0, 0, 0, 0, 0, 0)
datetext_default = '0000-00-00 00:00:00'
datetext_format = '%Y-%m-%d %H:%M:%S'
```

D.3 MailUtils

Help on module `webmessage_mailutils`:

NAME

`webmessage_mailutils`

FILE

`/home/cdsware/lib/python/cdsware/webmessage_mailutils.py`

DESCRIPTION

```
# -*- coding: utf-8 -*-
## $Id: webmessage_mailutils.py,v 1.2 2005/11/14 12:14:10 greg Exp $
##
## Some functions on html
##
## This file is part of the CERN Document Server Software (CDSware).
## Copyright (C) 2002, 2003, 2004, 2005 CERN.
##
## The CDSware is free software; you can redistribute it and/or
## modify it under the terms of the GNU General Public License as
## published by the Free Software Foundation; either version 2 of the
## License, or (at your option) any later version.
##
## The CDSware is distributed in the hope that it will be useful, but
## WITHOUT ANY WARRANTY; without even the implied warranty of
## MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
## General Public License for more details.
##
## You should have received a copy of the GNU General Public License
## along with CDSware; if not, write to the Free Software Foundation, Inc.,
```

59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.

FUNCTIONS

`email_quote_txt(text, indent_txt='>>', linebreak_input='\n', linebreak_output='\n')`

Takes a text and returns it in a typical mail quoted format, e.g.:

C'est un lapin, lapin de bois.

>>Quoi?

Un cadeau.

>>What?

A present.

>>Oh, un cadeau.

will return:

>>C'est un lapin, lapin de bois.

>>>>Quoi?

>>Un cadeau.

>>>>What?

>>A present.

>>>>Oh, un cadeau.

@param text: the string to quote

@param indent_txt: the string used for quoting (default: '>>')

@param linebreak_input: in the text param, string used for linebreaks
default: '

,

@param linebreak_output: linebreak used for output
default: '

,

@return the text as a quoted string

`email_quoted_txt2html(text, tabs_before=0, indent_txt='>>', linebreak_txt='\n', indent_html`

Takes a typical mail quoted text, e.g.:

hello,

you told me:

>> Your mother was a hamster and your father smelt of elderberries

I must tell you that I'm not convinced. Then in this discussion:

>>>> Is there someone else up there we could talk to?

>> No. Now, go away, or I shall taunt you a second time-a!

I think we're not going to be friends!

and return an html formatted output, e.g.:

hello,

you told me:

<div>

Your mother was a hamster and your father smelt of elderberries

</div>

I must tell you that I'm not convinced. Then in this discussion:

<div>

<div>

Is there someone else up there we could talk to?

</div>

No. Now, go away, or I shall taunt you a second time-a!

</div>

```

        I think we're not going to be friends!

    @param text: the text in quoted format
    @param tabs_before: number of tabulations before each line
    @param indent_str: quote separator in email (default: '>>')
    @param linebreak_str: line separator in email (default: '
')
    @param indent_html: tuple of (opening, closing) html tags.
                        default: ('<div class="commentbox">', "</div>")
    @param linebreak_html: line separator in html (default: '<br/>')
    @return string containing html formatted output

```

D.4 TextUtils

Help on module textutils:

NAME

textutils

FILE

/home/cdsware/lib/python/cdsware/textutils.py

DESCRIPTION

```

# -*- coding: utf-8 -*-
## $Id: textutils.py,v 1.1 2005/11/08 13:59:12 greg Exp $
##
## Some functions on text...
##
## This file is part of the CERN Document Server Software (CDSware).
## Copyright (C) 2002, 2003, 2004, 2005 CERN.
##
## The CDSware is free software; you can redistribute it and/or
## modify it under the terms of the GNU General Public License as
## published by the Free Software Foundation; either version 2 of the
## License, or (at your option) any later version.
##
## The CDSware is distributed in the hope that it will be useful, but
## WITHOUT ANY WARRANTY; without even the implied warranty of
## MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
## General Public License for more details.
##
## You should have received a copy of the GNU General Public License
## along with CDSware; if not, write to the Free Software Foundation, Inc.,
## 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.

```

FUNCTIONS

```
indent_text(text, nb_tabs=0, tab_str=' ', linebreak_input='\n', linebreak_output='\n')
    add tabs to each line of text
    @param text: the text to indent
    @param nb_tabs: number of tabs to add
    @param tab_str: type of tab (could be, for example "    ", default: 2 spaces
    @param linebreak_input: linebreak on input
    @param linebreak_output: linebreak on output
    @return indented text as string
```

