

Development of a new tool in e-Catalunya

Master's Degree in Information Technology Project

Director: Rosa M. Martín Santiago

Tutor: Jose María Barceló Ordinas

Lluís Suñol Juliachs
September 2008

1. INTRODUCTION	7
1.1. CONTEXT.....	8
1.2. FINAL MASTER PROJECT OBJECTIVES	9
1.3. THE E-CATALUNYA PLATFORM	11
1.3.1. <i>Portals</i>	13
1.3.2. <i>Groups</i>	15
1.3.3. <i>Members</i>	17
1.3.4. <i>Tools</i>	19
2. SEARCHING FOR THE NEW WIKI ENGINE	23
2.1. WHAT IS A WIKI?.....	24
2.2. WHAT IS OPEN SOURCE?	26
2.3. STUDY OF THE CURRENT "XWIKI" ENGINE	28
2.4. SEARCH AND ANALYSIS	31
2.4.1. <i>Engines which were directly discarded</i>	32
2.4.2. <i>Engines which matched the minimum requirements</i>	38
2.4.3. <i>Decision</i>	47
3. REQUIREMENT ANALYSIS	49
3.1. FUNCTIONAL REQUIREMENTS.....	50
3.1.1. <i>E-Wiki features</i>	51
3.1.2. <i>E-Wiki tool in e-Catalunya</i>	58
3.2. NON-FUNCTIONAL REQUIREMENTS	61
3.2.1. <i>e-Catalunya platform independent modelling</i>	62
3.2.2. <i>Visual integration</i>	63
3.2.3. <i>Accessibility</i>	64
3.2.4. <i>Usability</i>	65
3.2.5. <i>Reliability</i>	66
3.2.6. <i>Security</i>	67
4. SPECIFICATION	69
4.1. ACTORS.....	70

4.2.	USE CASES	72
4.2.1.	<i>Assign a new e-Wiki to a group or a portal.....</i>	73
4.2.2.	<i>Close an e-Wiki</i>	75
4.2.3.	<i>Modify the properties of an e-Wiki tool.....</i>	76
4.2.4.	<i>Delete an e-Wiki tool</i>	77
4.2.5.	<i>View a document.....</i>	78
4.2.6.	<i>Export a document</i>	80
4.2.7.	<i>View the history of a document.....</i>	81
4.2.8.	<i>View an old version of a document</i>	82
4.2.9.	<i>Create a document</i>	83
4.2.10.	<i>Edit a document</i>	85
4.2.11.	<i>Upload a file.....</i>	90
4.2.12.	<i>Revert a document</i>	92
4.2.13.	<i>Clear the history of a document.....</i>	93
4.2.14.	<i>Unlock documents</i>	94
4.2.15.	<i>Rename documents</i>	95
4.2.16.	<i>Remove documents</i>	97
4.2.17.	<i>Restore documents</i>	99
4.2.18.	<i>Delete documents.....</i>	101
4.2.19.	<i>Comment a document.....</i>	103
4.2.20.	<i>Edit a comment.....</i>	105
4.2.21.	<i>Delete a comment</i>	107
5.	DESIGN AND IMPLEMENTATION	109
5.1.	TECHNOLOGICAL ENVIRONMENT	110
5.1.1.	<i>Web servers</i>	111
5.1.2.	<i>eXo Platform: portal framework manager.....</i>	113
5.1.3.	<i>Programming languages.....</i>	114
5.1.4.	<i>Data management.....</i>	118
5.2.	ARCHITECTURE.....	119
5.3.	CONCEPTUAL MODEL	122
5.4.	FINAL RESULT	130
5.5.	DEVELOPMENT ENVIRONMENT	140
5.6.	RUNNING ENVIRONMENTS	142
5.7.	DEVELOPMENT, TESTING AND DEPLOYMENT	145

6. PROJECT CONCLUSIONS	147
6.1. COST ANALYSIS.....	148
6.2. FUTURE IMPROVEMENT PROPOSALS.....	149
6.3. FINAL MASTER PROJECT CONCLUSIONS	151
6.4. PERSONAL CONCLUSIONS.....	152
6.5. ACKNOWLEDGES	154
7. BIBLIOGRAPHY	155



1. Introduction

This chapter will introduce both the context where this Final master Project (FMP) takes place which is e-Catalunya and its objectives.

An overall brief description of the context in which this final master project has been developed follows.

1.1. Context

This project has been developed in the Computing Laboratory of the *Barcelona School of Informatics (LCFIB)*. The main objective of this project is to develop a new tool for the e-Catalunya platform.

e-Catalunya Platform is a collaborative project between the *Generalitat de Catalunya* and the *Barcelona School of Informatics*. e-Catalunya consists of a collaborative web-based framework developed by the *LCFIB* at the *Barcelona School of Informatics*. The purpose of e-Catalunya is to enhance better communication between different citizen sectors such as health, justice or teaching personnel by providing them many ways to interact. To do so, e-Catalunya provides them with tools they can use to interact and contribute with their own.

e-Catalunya platform is currently developed by a team of six developers working on different roles: analysis and design, programming, testing and maintenance. This team does a weekly meeting in order to check the evolving of the whole platform, define new working lines and distribute tasks.

Later on we will see a little introduction to e-Catalunya.

1.2. Final master project objectives

The main objective of e-Catalunya established in its beginnings was to be born as the result of a search for open source web tools, such as the ones mentioned before (blogs, wikis, calendars, etc.).

That search resulted in a list of open source web tools specially selected to be easily integrated all together in a single web platform. This means that researchers needed to analyze all the available technologies and to select those who seemed to have more probabilities to fit better all together in the platform, giving to user the feeling that he is using a framework specially designed for him, instead of one made of different puzzle pieces stuck at force.

One of those searched, studied, analyzed and finally selected technologies was an open source tool named "[XWiki](#)", in its beta version 0.9.543.

This technology was (and it still is) an open source content management system (CMS) the features of which fitted better with the needs of the whole e-Catalunya project in its beginnings.

The XWiki engine has many advantages. The most important is that it does provide an easy and extensible engine which can be used to create new tools. This engine was initially used to create the calendar, the blog, the file repository and the album. It was also supposed to be easily to integrate with the rest of the software.

In contrast, this engine also has some weaknesses. The main problem is that this engine is very slow and resource consumer. This means than it can take a long time between a request from a user and its server response. In addition, at the time it was studied and integrated to e-Catalunya, this engine was on a beta testing phase. This disadvantage was underestimated, expecting the tool to not to carry many errors to the platform and expecting the platform to be able to manage them all. As this software is large and complex,

errors are hard to solve, because they are not easy to detect, find, and debug.

When all of the starting objectives of e-Catalunya were finished new user requirements were determined. Some of these requirements were to substitute those tools which were managed by the XWiki engine by new ones more efficient and with richer functionalities.

The first tool which was substituted was the Blog tool. The rest of the tools using the XWiki engine have been being replaced gradually.

The main objective of this FMP is to substitute the last one of those tools mentioned which is still remaining in the platform and has not been substituted: **the wiki tool**. The result of this project will be a new wiki engine ready to be used by e-Catalunya members.

A wiki is a collection of web pages designed to enable anyone who accesses it to contribute or modify content.

This main objective is divided in minor goals. First of all, we will have to search for open source wiki engines and decide which of them is more suitable to substitute the wiki engine.

Now an introduction to the e-Catalunya platform follows.

1.3. The e-Catalunya platform

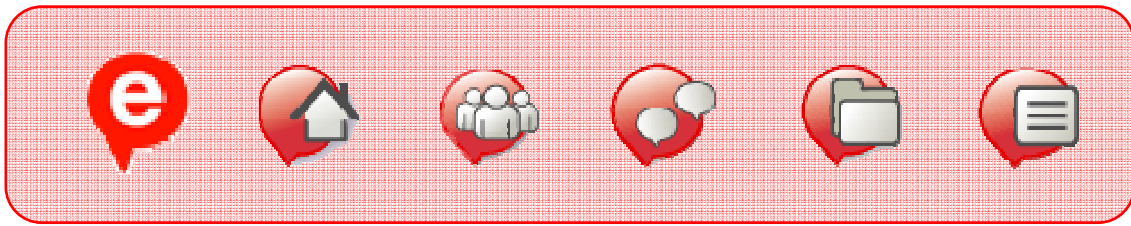


Figure 1: e-Catalunya iconography

e-Catalunya consists of a collaborative web-based framework. The whole e-Catalunya platform is made of portals, groups, members and tools. It has a hierarchical structure.

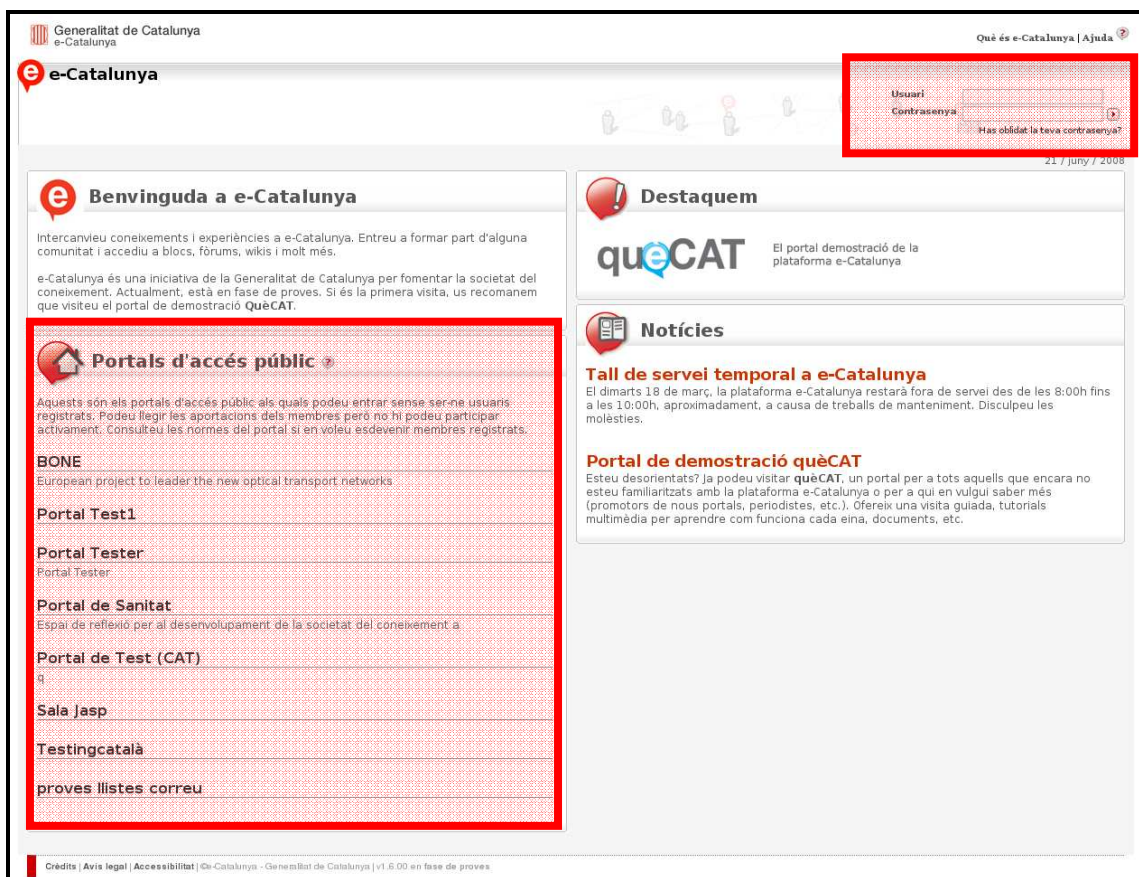


Figure 2: starting page of e-Catalunya where we can see the login box and portals of public access

The top of this hierarchy is made of portals. Members in e-Catalunya are grouped by groups, and these groups belong to portals as shown in Figure 3.

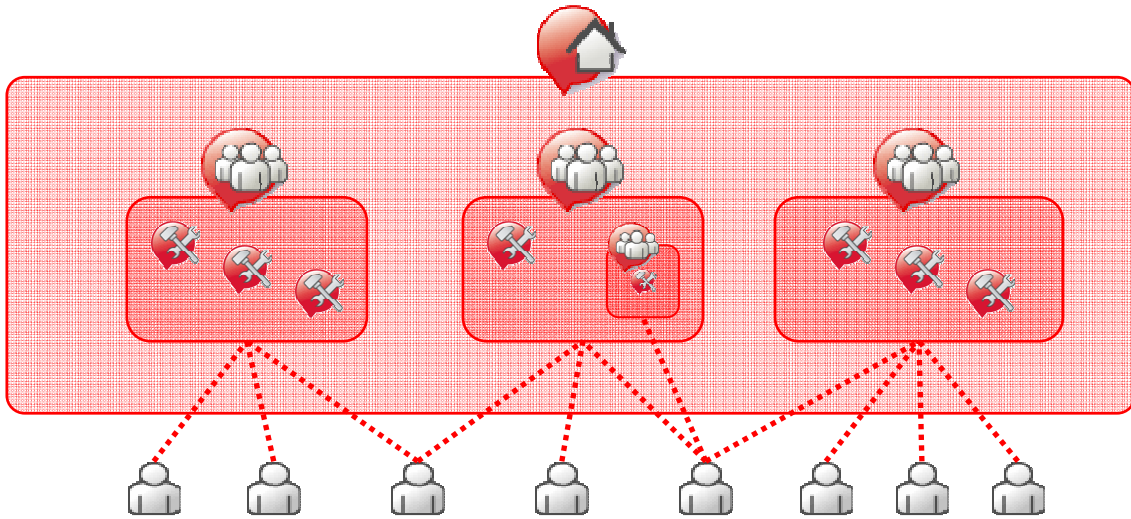


Figure 3: Hierarchical schema of e-Catalunya

1.3.1. Portals

Portals are the top of the hierarchical organization of e-Catalunya. There are 47 current active portals in this platform, each one completely independent from each other.



Figure 4: Portals are represented with this icon in e-Catalunya

Portals can be either public or private. Portals are made of groups, members and tools. Each portal is based on a main topic and all of the groups, members and tools, which this portal is made of, are related by this topic. The objective of a portal is to relate large collectives of people that may have never met.

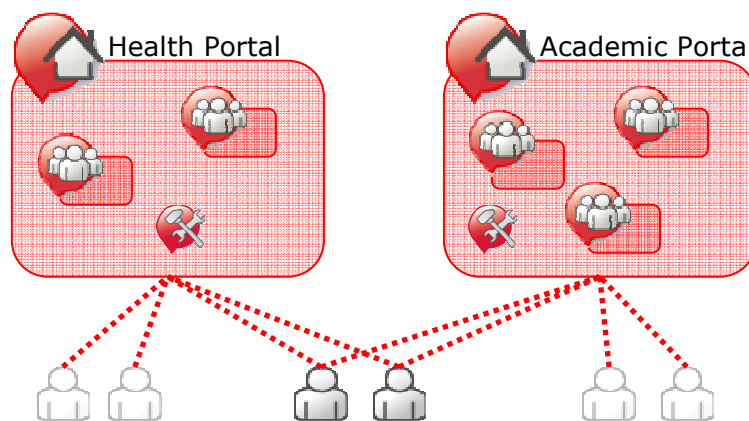


Figure 5: different portals can share members, but they are independent one from each other.

Members of a portal can contribute in the tools assigned to the portal. The e-Catalunya platform attempts to match members that are similar in each single portal by analyzing its behaviour. This analysis results in a knowledge network which is used to suggest documents that may be useful for members.

In example, some of the active portals in e-Catalunya are:

- Health portal
- Justice portal
- Development portal

The screenshot shows the e-Catalunya website interface. On the left, there is a sidebar titled "Els meus portals" containing a long list of various portals and services. On the right, there is a main content area titled "Novetats als meus portals" which displays news updates from different departments like AQU Catalunya, Agència de Protecció de la Salut, etc.

Generalitat de Catalunya
e-Catalunya

Què és e-Catalunya | Ajuda

e-Catalunya

Benvingut, Administració e-Catalunya
Tancar sessió

4 / setembre / 2008

Els meus portals

- SS Qualitat i millora
- AQU Catalunya
- Administradors
- Agència de Protecció de la Salut
- Atenció Ciutadana
- Atenció Primària de Barcelona
- Barris amb projectes
- Catalunya exterior
- Cluster Innovació
- Commemoracions
- Comunitat Virtual del Servei d'Ocupació de Catalunya
- Consell Nacional de Dones de Catalunya
- Coordinació Interdepartamental
- Direcció General de Planificació i Avaluació
- Distribució ECAI Platform
- Durs: Pla de serveis i continguts
- EAPC
- Fòrum del temps de Treball
- GTS (Governs Territorials de Salut)
- Generalitat Girona
- Gestió Documental i Arxius
- Igualtat d'oportunitats en el treball
- Institut Català de les Dones
- Internet del Futur
- Llengua catalana
- Participació Salut
- Portal Euroregió Pirineus Mediterrània
- Portal Societat del Coneixement
- Portal d'Acció Social i Ciutadania
- Portal d'Economia i Finances
- Portal d'Educació
- Portal de Justícia
- Portal de Participació Ciutadana
- Portal de Sanitat
- Portal de Treball
- Portal de creació d'empreses
- Portal desenvolupament
- Portal i2CAT
- Reducció de barreres a l'activitat econòmica
- SITIC Salut
- Secretaria de Relacions Institucionals i Participació
- T-Govern
- TICSalut
- Talla amb els mals rotllos
- Taulas de debat del Pacte Nacional per a la Recerca i la Innovació
- Unidiscat
- Universitats
- XBEG Xarxa de Biblioteques Especialitzades de la Generalitat
- Xarxa Transversal
- Xarxa de Biblioteques del Sistema Sanitari Públic de Catalunya
- e-Difusió Artística
- queCAT
- Àmbit de Regulació Autònoma

Novetats als meus portals

AQU Catalunya

- Calendari (Àrea de Matemàtiques i Ciències Experimen...)
Enviar ficha (09/09/2008)
Rexachs del Rosario, Dolores - 16/07/2008
- Calendari (Àrea de Matemàtiques i Ciències Experimen...)
Reunión del grupo área de matemàtica i ciènc...
Rexachs del Rosario, Dolores - 16/07/2008
- Calendari (Formació en Ciències de l'Activitat Físic...)
Reunió GUIA nº 13 (25/06/2008 18:00)
Sebastià Obrador, Enric Maria - 15/06/2008

Agència de Protecció de la Salut

- Fòrum Gestió del Coneixement (Gestió del coneixement)
Creatió d'un espai per les CoP dins del Grups...
Colomer Giner, Antoni - 10/08/2008
- Fòrum Gestió del Coneixement (Gestió del coneixement)
Creatió d'un espai per les CoP dins del Grups...
Rigau Pellissà, Blanca - 07/08/2008
- e-Calendar (Gestió del coneixement)
4t matí d'innovació - Tecnologies de la infor...
Colomer Giner, Antoni - 31/07/2008

Atenció Ciutadana

- Documents (Direcció del projecte)
Revisió i Pla d'Accions 08-09.pdf
Sanfeliu, Manon - 03/09/2008
- Documents (Direcció del projecte)
OCmandaments juliol.pdf
Sanfeliu, Manon - 02/09/2008
- Documents (Direcció del projecte)
Seguiment Catalunya 20080903.doc
Puente Lumbrarres, Hector - 02/09/2008

Atenció Primària de Barcelona

- e-Calendar (Estudi Grups Psicoeducatius Depressió)
Reunió Grup Investigació Grups Psicoeducatius...
Casañas Sánchez, Rodo - 12/06/2008
- e-Calendar (Curs de Disseny de Projectes de Recerca)
Mòdul 5. La formació de l'equip investigador...
LT, Anna - 05/06/2008

Barris amb projectes

- Repositori (Xarxa de barris)
17 Recull de premsa 22 al 28 agost 2008.pdf
Vidal guixens, Mª Eugènia - 02/09/2008
- Repositori (Xarxa de barris)
Hospital de Llobregat - Florida i Pubilla C...
Obiols, Maria - 02/09/2008
- Repositori (Xarxa de barris)
Cornellà .pdf
Obiols, Maria - 02/09/2008

Catalunya exterior

- Contacteu amb catalans/es a Àsia i Oceania... (...)
El meu camí cap Austràlia
Gallench, Oscar - 30/08/2008
- Contacteu amb catalans/es a Àsia i Oceania... (...)
El meu camí cap Austràlia
Busch Biniques, Sara - 30/07/2008
- Contacteu amb catalans/es a Àsia i Oceania... (...)
El meu camí cap Austràlia
Viñas Baron, Josep - 28/07/2008

Cluster Innovació

- e-Calendar (Turisme)
Reunió general de seguiment (23/07/2008 11:00...
Giménez Esteban, Rafael - 17/07/2008
- Fòrum (Turisme)
Comentaris sobre l'anàlisi funcional
Montero Garcia, Jordi - 07/07/2008
- Fòrum (Turisme)
Comentaris sobre l'anàlisi funcional
Giménez Esteban, Rafael - 30/06/2008

Comunitat Virtual del Servei d'Ocupació de Catalunya

- Fòrum (Projecte Sistema de gestió d'espera a les ofi...)
nou sistema de gestió d'espera
Serrat Maipo, Just - 22/08/2008
- Fòrum (Projecte Sistema de gestió d'espera a les ofi...)
nou sistema de gestió d'espera
Ramos Ortiz, Antoni - 21/08/2008
- Fòrum (Projecte Sistema de gestió d'espera a les ofi...)
nou sistema de gestió d'espera
Serrat Maipo, Just - 20/08/2008

Coordinació Interdepartamental

- Agenda Territori (Territori)
"Agua y Tierra: Gestión Territorial y Foresta...
Mateu Llevadot, Xavier - 09/06/2008
- Agenda Territori (Territori)
"Agua y Tierra: Gestión Territorial y Foresta...
Mateu Llevadot, Xavier - 09/06/2008
- Agenda Territori (Territori)
conferencia internacional Agua i Boscos (30/...

Figure 6: My portals (public and private)

1.3.2. Groups

Groups are the middle stage of the hierarchical organization of e-Catalunya. There are about 750 active groups in e-Catalunya.



Figure 7: Groups are represented with this icon in e-Catalunya

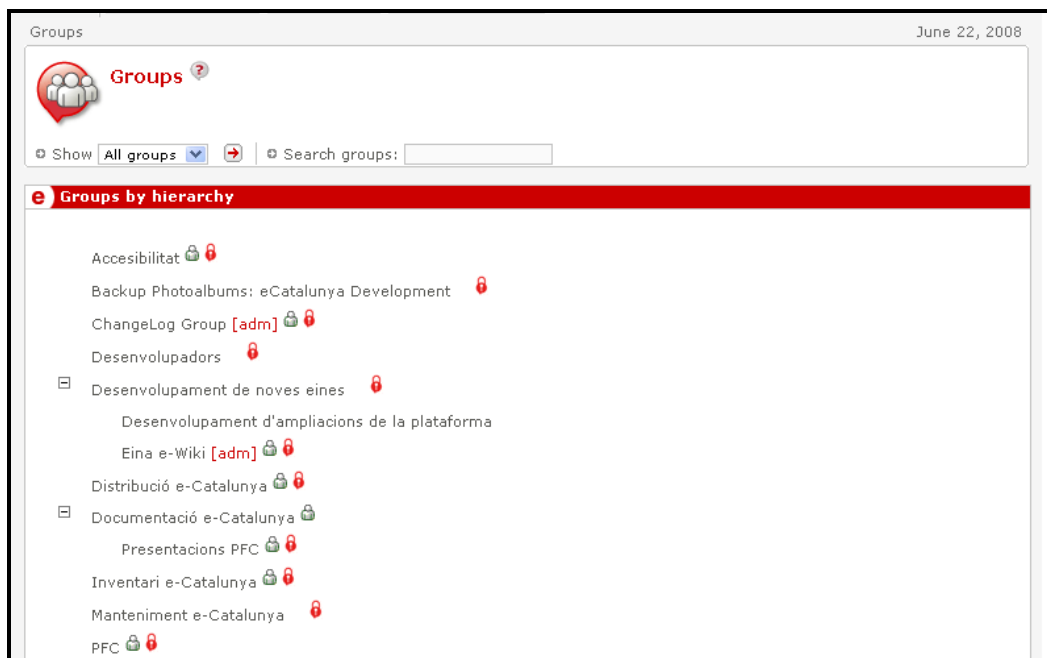


Figure 8: hierarchical list of groups of a portal

Though users can be assigned directly to a portal and interact with other portal members and tools, its more common for users to be associated to groups which belong to a concrete portal. Users, then, interact with the tools assigned to their groups.

Groups can be nested. This means that groups are made of tools, members and also other groups (usually called "subgroups"). The objective of groups is to relate more specific collectives than the portal ones. Tools that belong to groups will probably be closer to its members than the portal ones due to this reason.

As an example, there could be a Health Portal in e-Catalunya dedicated to health related activities in e-Catalunya.

There could be groups like a "Health Personnel of Hospital de Sant Pau", where there could be general practitioners, surgeons, supervisors, cleaning staff, or anyone who works in this hospital. The objective of this group would be to establish a meeting point for general issues related to the hospitals behaviour. Also following with the example, this group could also contain another group called "Basketball team of Hospital de Sant Pau", dedicated exclusively to the basketball team made by the personnel of this hospital. In this second group there could be a calendar tool, where they could publish match days of the basketball team.

The objective of both groups is completely different, but the main topic is maintained: both groups relate health personnel of a hospital.

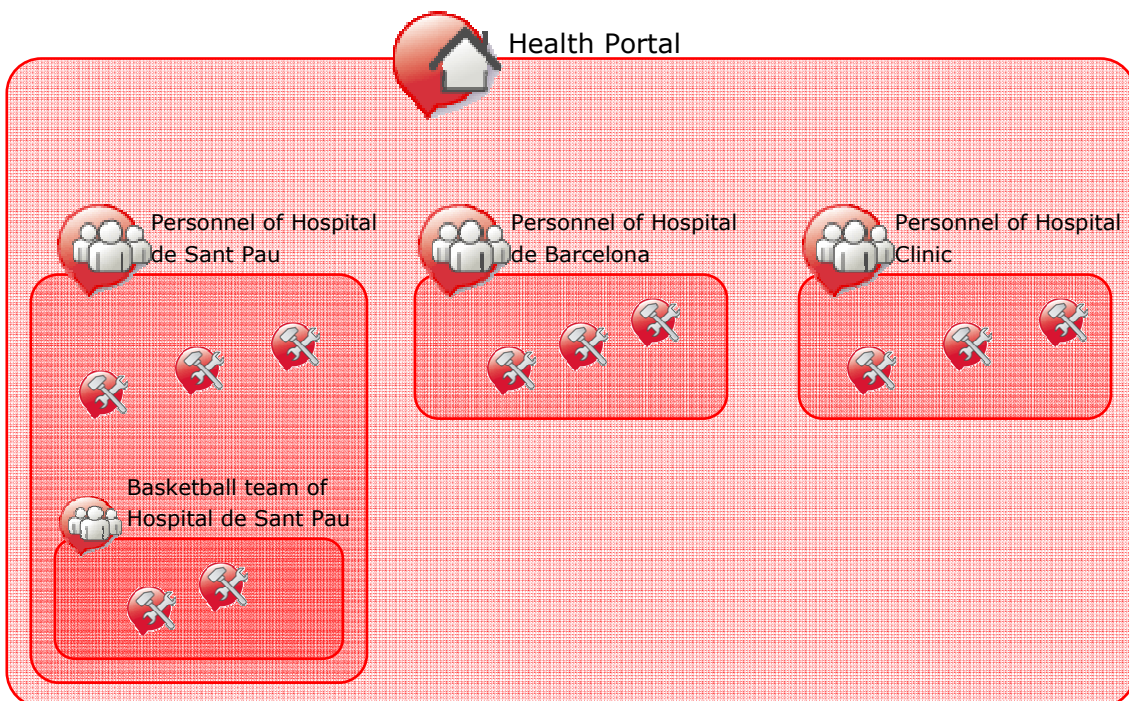


Figure 9: example of a possible group hierarchy inside the Health portal

1.3.3. Members

Members are the base of the hierarchical organization of e-Catalunya. There are about 11.500 active members in e-Catalunya.

Members of e-Catalunya are related to portals and groups. It is mandatory that if a member of e-Catalunya belongs to a group, this member also belongs to the portal in which this group is placed. A member of e-Catalunya can be related to more than one group. This implies that a member can be related to more than one portal. There is not any restriction about how many groups a member can belong to. Members can have different roles on their groups. There are three basic roles in e-Catalunya:

- **Member:** a member of a group or a portal can usually contribute with his own to the tools of the group or portal he belongs.
- **Moderator:** a moderator of a group or a portal may contribute, but also moderate contributions of regular members, by editing or even banning unwanted content.
- **Administrator:** an administrator of a group can perform administrative tasks, such as management of groups, tools, memberships, etc.

These roles represent the level of access to the platform. These roles can be applied, in general to two different scopes:

- **Group:** when someone is assigned to a group, is implicitly assigned to the portal in which this group is placed. That's why a portal membership does not give much responsibility. It is placed just above guests.
- **Portal:** In the opposite, a portal administrators and moderators are also administrator or moderators of all the groups of that portal. Only the platform administrator has more privileges, that is why they are placed just below him.

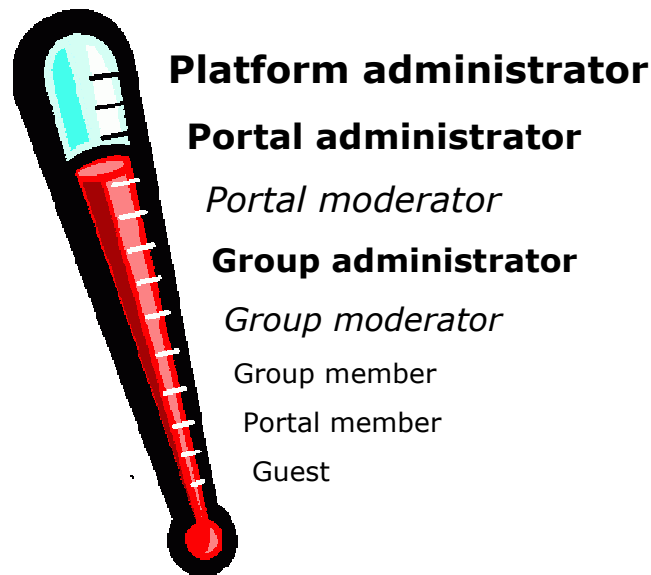


Figure 10: Ranking of general access by role and scope

As shown in Figure 10, there are two roles not mentioned before:

- **Guest:** a guest is anyone who does not belong to any portal or group. Everyone can access as guest only by not logging in at the platform. Guests will only be able to access to portals, groups, and tools which are public. Guests only can contribute to public tools which are configured to permit anonymous contributions.
- **Platform administrator:** this special role is not really a role, because only one member can play this role: the administrator. This member not only has access to every portal, group and tool, but also performs many administrative tasks of the entire platform.








1.3.4. Tools

As mentioned before, members in e-Catalunya can use tools to contribute, publish, advertise, share contents, etc. Currently those groups mentioned before have a total of 3.900 instanced tools.



Figure 11: Tools (in general) are represented by this generic icon, but each kind of tool has his own icon.

These available tools are, up to date, the following:

-  Photo albums: this tool permit members to share images, photos, schemas and many types of static images in a kind of album look format.
-  Blogs: this tool is a kind of personal diary which lets members to add entries, making them available for other members who will be able to read and comment them.
-  Calendars: this tool let members publish and announce dated activities.
-  File repositories: uploading and downloading any kind of files.
-  Forums: this tool is used to make popular discussions.
- Mailing lists: email based group messaging tool.
- RSS: Syndication tool.
-  Mailrooms: this is an e-Catalunya specific tool. It is a kind of survey engine.
-  Wiki: popular collaborative web-content publishing tool.
- Bug tracker: reporting system bugs tool.

The main purpose of all this tools is to enhance better communication between e-Catalunya members.

As soon as an administrator of a group adds many of these tools to the group, members can start interacting with them. There is

no restriction about how many tools can be added to a group. Groups can even owe many tools of the same type as needed.

Usually tools are assigned to groups, but this rule is not mandatory. Many of the tools mentioned before can belong not only to groups, but also to particular users, like blogs or file repositories. Also, since version 1.6 of e-Catalunya, tools can also be assigned to portals.


One of the objectives of e-Catalunya is to analyze member's behaviour, compare them all and match those members who have similar activities.





Figure 12: Example of the knowledge network of a portal


This knowledge is finally used to recommend to each member which available documents may interest him.


Recommended documents

 **Errors/Bugs** (*Suggerencies/Errors* ...)
EspaiPersonal
Alba\$ Coll Triginer - 26/01/2007

 **Errors/Bugs** (*Suggerencies/Errors* ...)
ReportsIdioma
Alba\$ Coll Triginer - 26/01/2007

 **Errors/Bugs** (*Suggerencies/Errors* ...)
Mails
Lucas Ponce Liu - 07/02/2007

 **Errors/Bugs** (*Suggerencies/Errors* ...)
AlbumDImatges
Alba\$ Coll Triginer - 23/10/2007

 **Errors/Bugs** (*Suggerencies/Errors* ...)
EinesDeGruppersonal
Alba\$ Coll Triginer - 26/01/2007



 **Subscriptions**  **RSS**

Figure 13: Example of recommended documents of a portal



2. Searching for the new wiki engine

In this chapter we will introduce some basic concepts, such as “wiki” or “open source”.

Afterwards, we will take a look at the old wiki engine used in the e-Catalunya platform before this FMP ended.

Then we will see all of the wiki engines studied in this phase of the project in order to replace the used one to finally end with the final decision about that.

2.1. What is a wiki?

Before starting the research for the new wiki engine, let's define what a wiki is.

By definition, a wiki is a collection of web pages designed to enable anyone who accesses it to contribute or modify content.

In our particular case we do not want everyone all over the world to be able to access and modify content. We will enjoy having a little bit more privacy than this. We will argue this later.

Wikis usually are scoped to a main topic. There are many well known wikis available through the internet with many different topics.

In example, Wikipedia (The Free Encyclopedia) is probably the biggest wiki on the internet and its objective is to contain information as encyclopaedias do.

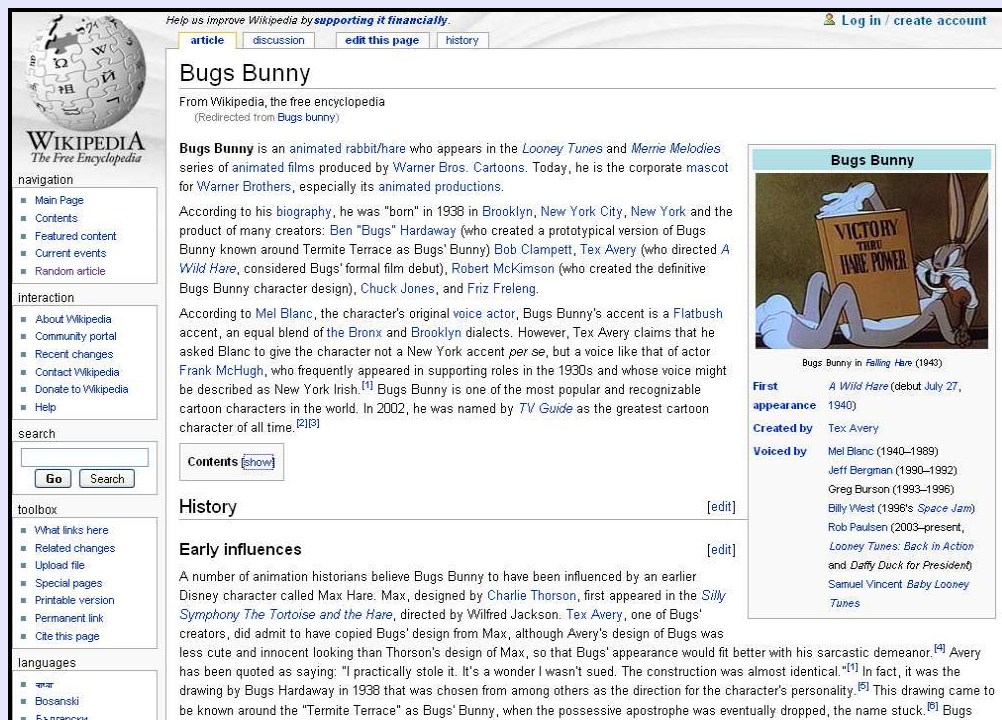


Figure 14: image of a random entry on the Wikipedia

There are other wikis not so well known, such as Wikitravel, which its main topic is to give information about

travels, or Wookipedia, which contains information about Star Wars.

But we do not want to have a unique instance of a wiki in which the users of e-Catalunya can edit their entries. What we want is to make this engine available to the final users of e-Catalunya so they can create as many instances of wikis as they want with the topics they want, being independent ones from each others.

It would be an error to try to compare the new wiki engine to the Wikipedia. The new wiki engine is not an instance of a wiki. It should be compared to the MediaWiki, which is the engine used by the Wikipedia to host and manage its entries.

2.2. What is open source?

Open source is a development methodology, which offers practical accessibility to a product's source (goods and knowledge).

As mentioned before, the philosophy of e-Catalunya is to integrate open source tools. Why doing this? Open source tools have some advantages and disadvantages.

The main advantage of open source (in general) is that this kind of software is usually used and debugged by large communities. As they are open, everybody is free to analyze and modify its source code, and also publish the changes. This means that by integrating an open source wiki engine we will probably obtain a hardly tested and well working tool.

The main disadvantage of open source, in the particular case of integrating a wiki engine, is that it may have fewer features than the minimum required, or it may have too many useless features or both. Another disadvantage of open source is that its integration is not trivial. It has to fulfil some qualities, such as similar appearance or similar behaviour to the rest of the platform, etc.

For example, it's expected for the new wiki engine to have the same look as the rest of the platform. This includes not only colours, font types or sizes, but also structure of displayed information, feedback to the final user.

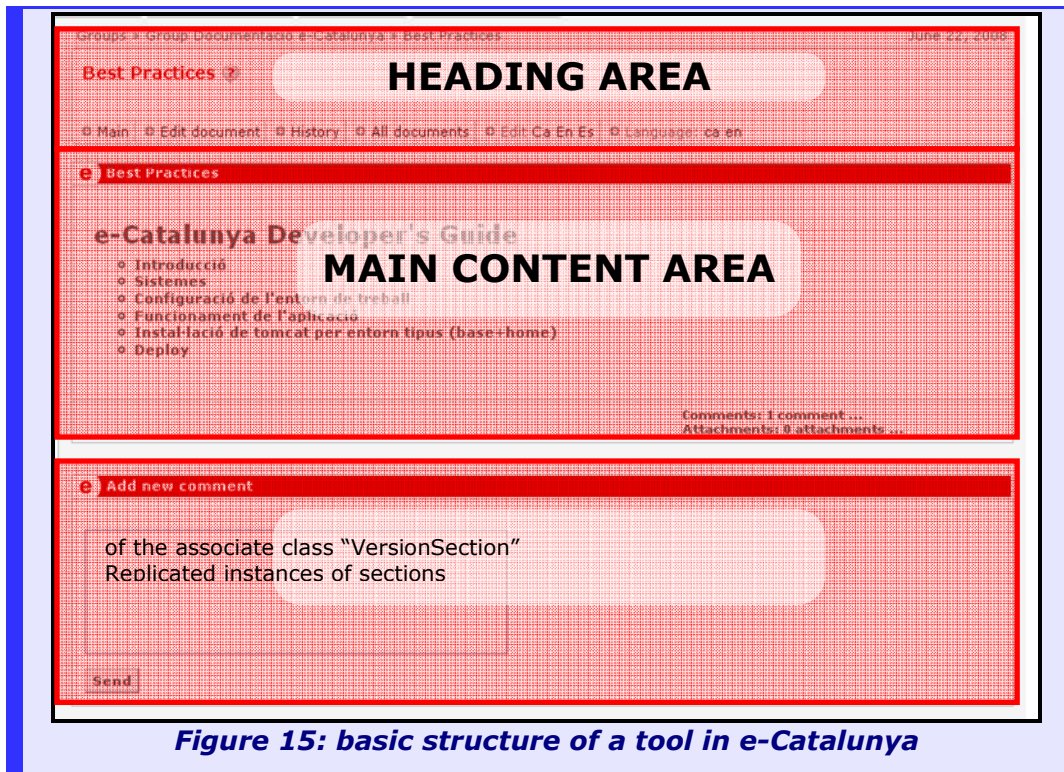


Figure 15: basic structure of a tool in e-Catalunya

Once explained what a wiki is and what open source is, we can go on with the objective of searching for the new wiki engine.

In the present, there is a wiki engine available based on an adaptation of the XWiki 0.9.543. The new wiki engine had to provide at least the same services as the old one did. This was mandatory because we did not want to inconvenience the customer. We wanted to let users to be able to do the same. That's why, before searching for open source wiki engines, we better start by studying the features offered at present.

2.3. Study of the current “XWiki” engine

As mentioned before, the wiki engine used in Catalunya is part of bigger software, which is the XWiki, adapted and integrated into e-Catalunya platform. As one of the principal objectives of the new wiki engine was to bring, at least, the same features offered by the current one, we have had to study it.

This stage of the project finished with a list of minimum requirements for the new wiki engine. In addition, another list with desired requirements was also elaborated.

Features offered by the current wiki engine:

- Create public and private wiki instances.
- Create and edit the starting wiki page.
- Create and edit unlimited amount of wiki pages.
- View and edit pages in the English, Spanish and Catalan languages.
- View a history of a page and compare different versions of it graphically.
- List all the available documents of a wiki.
- Edit pages in a specific wiki syntax.
- Insert some HTML tags.
- Comment pages.
- Add attachments.

These features were mandatory for the new engine. In addition, it was desirable for the new wiki engine to achieve some other features. These desirable features can be classified into two different types:

Developer oriented features

These features are useful for developers. They talk about how the new engine will be implemented and which technologies will use.

Final users of the new tool will not notice about these features directly. It's important for the new wiki engine to:

- To have a GPL compatible license, as the e-Catalunya platform is distributed with this license.
- To be written in Java or PHP (both already used in the platform).
- To work over a MySQL database management system (already used in the platform).
- To have access control lists to be able to adapt the different roles available in the e-Catalunya platform.
- To have an authentication backend such as LDAP or JOSSO (both used in e-Catalunya).
- To accept HTML tags in the content of the pages. We will see this later on.

User oriented features

User oriented features are those which the engine offers directly to the end user which will be creating, editing and managing the wiki. They are:

- To enable a preview of the document before saving it.
- To have a dynamic index page.
- To have a concurrency conflict handling mechanism.
- To enable editing at a section level, without needing to lock entire pages.
- Not to use a special markup language to write pages (to use a WYSIWYG ¹editor).

¹ WYSIWYG is an acronym for "What You See Is What You Get", used in computing to describe a system in which content displayed during editing appears very similar to the final output.

- To offer a printable version of documents.
- To have an automated table of contents system.
- To export to different formats (PDF, XML, etc.)
- To have a file attachment system (if none is available, it would be used the same which was used in the old wiki engine).
- Finally the new engine would have to be easy to integrate into the platform, easy to update, reliable and scalable in number of members using it and in number of documents stored.

The next step was to search for open source developed wiki engines available through the internet.

2.4. Search and analysis

This phase started with a general overview of the current state of wiki engines available. We studied the most important and the most used wiki engines. Many of them are widely used all over the internet and some others are more experimental.

Here is the list of wiki engines which were studied:

- DokuWiki
- MediaWiki
- TWiki
- MoinMoin
- PmWiki
- PhpWiki
- TikiWiki
- Daisy
- XWiki

Some of these engines were discarded immediately due to incompatibilities or a lack of some required features and some others were studied more thoroughly.

2.4.1. Engines which were directly discarded

In this section there's the list of discarded open source wiki engines and the reasons why they were discarded.

DokuWiki

DokuWiki is a standards-compliant, simple-to-use wiki which allows users to create rich documentation repositories. It provides an environment for individuals, teams and companies to create and collaborate using a simple yet powerful syntax that ensures data files remain structured and readable outside the wiki. Unlimited page revisions allows restoration to any earlier page version and with data stored in plain text files, no database is required. A powerful plug-in architecture allows for extension and enhancement of the core system.

Here is the list of requirements for this engine:

General Features

Last Release	26/06/2007	✓
License	GPL 2	✓
Programming Lang.	PHP	✓
Data Storage	Files	✗
Page permissions	Yes	✓
ACL	Yes	✓
Authentication backend	Text file, LDAP, MySQL	✓

Usability

Preview	Yes	✓
Page history	Yes	✓
Revision differences	To current	✓
Page index	Yes	✓
Interface languages	English, Spanish and Catalan	✓
Page redirection	Plug-in	✓
Conflict handling	Page locking	✓
Section editing	Yes	✓
Double click edit	No	✓
WYSIWYG	Plug-in	⚠
HTML Tags	Optional	✓
Access keys	Yes	✓

Output

Printer Friendly	Print CSS	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Yes	✓

XML export	No	✖
PDF export	No	✖

Media

File attachments	Yes	✔
Embedded flash	Yes	✔
Embedded video	Plug-in	✔
Syntax	Different as the current syntax	!

- ✔ It is Ok
- ! Can cause some inconveniences
- ✖ Problem that can not be solved

Table 1: features of the DokuWiki engine

As we can see in Table 1, this wiki engine uses files to store the content of all the pages. This is the main reason for what it was directly discarded. Using files as data storage is slower than using a relational database management system. In addition, database managements systems are easily *indexable*.

What we mean by "indexable" is the capability of the content to be analyzed and indexed. To create an index of a document the only way is to read it entirely and extract those most important words.

Indexes are useful, for instance, when trying to search for some specific key words. A good indexation gives better search results.

MediaWiki

MediaWiki is a web-based wiki software application used by all projects of the Wikimedia Foundation, all wikis hosted by Wikia, and many other wikis, including some of the largest and most popular ones. MediaWiki is written in the PHP programming language, and can use either the MySQL or PostgreSQL relational database management system. MediaWiki is distributed under the terms of the GNU General Public License while its documentation is released under the GFDL and partly in the public domain, making it free and open source software.

MediaWiki is the engine used in the Wikipedia, the free Encyclopedia.

Here is the list of requirements for this engine:

General Features		
Last Release	10/09/2007	✓
License	GPL	✓
Programming Language	PHP (with OCaml and C)	✗
Data Storage	MySQL	✓
Page permissions	Yes	✓
Access control lists	Yes	✓
Authentication backend	Yes	✓
Usability		
Preview	Yes	✓
Page history	Yes	✓
Revision differences	Between all	✓
Page index	Yes	✓
Interface languages	Over 100	✓
Page redirection	Yes	✓
Conflict handling	conflict resolution	✓
Section editing	Yes	✓
Double click edit	Optional	✓
WYSIWYG editor	Plug-in	!
HTML Tags	Some	✓
Access keys	Yes	✓
Output		
Printer Friendly	Print CSS	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Yes	✓
XML export	Yes	✓
PDF export	Optional	✓
Media		
File attachments	Yes	✓
Embedded flash	Plug-in	✓
Embedded video	Plug-in	✓
Syntax	Different as the current syntax	!

Table 2: features of the MediaWiki engine

This engine presents many problems. The most important of them and, also, the reason for what it was discarded is that some parts of the engine are written in the OCaml and C programming languages.

This would require adapting the platform to accept these programming languages. This is not an easy task, because we

would have to adapt not only one environment, but all of the required technologies in order to make it work properly on the final production environment.

Another minor problem is that does not offer a built-in visual editor. A visual editor and the exporting to PDF features can be updated by installing some additional plug-ins. This can present problems make us expend extra time.

TWiki

TWiki is a flexible, powerful, and easy to use enterprise wiki, enterprise collaboration platform and knowledge management system. It is a Structured Wiki, typically used to run a project development space, a document management system, a knowledge base, or any other groupware tool, on an intranet or on the internet. Web content can be created collaboratively by using just a browser. Users without programming skills can create web applications. Developers can extend the functionality of TWiki with Plug-ins. TWiki fosters information flow within an organization; lets distributed teams work together seamlessly and productively; and eliminates the one-webmaster syndrome of outdated intranet content.

TWiki is installed on many web sites. Companies such as Motorola or SAP use the TWiki platform.

Here is the list of requirements for this engine:

General Features		
Last Release	03/03/2007	✓
License	GPL	✓
Programming Language	Perl	✗
Data Storage	Files	✗
Page permissions	Yes	✓
Access control lists	Yes	✓
Authentication backend	LDAP	✓
Usability		
Preview	Yes	✓
Page history	Yes	✓

Revision differences	Between all	✓
Page index	Yes	✓
Interface languages	14	✓
Page redirection	Yes	✓
Conflict handling	Conflict resolution	✓
Section editing	Plug-in	!
Double click edit	Plug-in	✓
WYSIWYG editor	Yes	✓
HTML Tags	All	✓
Access keys	Yes	✓
Output		
Printer Friendly	Print View	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Yes	✓
XML export	Plug-in	!
PDF export	Plug-in	!
Media		
File attachments	Yes	✓
Embedded flash	Plug-in	✓
Embedded video	Plug-in	✓
Syntax	Different as the current syntax	!

Table 3: features for the TWiki engine

This engine was rapidly discarded due to many reasons. The first one is the programming language it uses, which is Perl. Perl is another programming language such as PHP or Python. This programming language is not yet supported by the platform. In addition, this engine uses files to store all the data. We saw the reason for this problem in the DokuWiki section.

MoinMoin

MoinMoin is a wiki engine implemented in Python, initially based on the PikiPiki² wiki engine. Distributed under the terms of the GNU General Public License, MoinMoin is free software.

² PikiPiki is a small wiki engine written in Python which its objective is to keep as small as possible. The current version is about 596 lines of commented code.

The feature set of fine grained access control, simple user groups, visual editor, easy install, simple but efficient spam protection, easy theming combined with a simple code base makes it often the wiki of choice for many open source projects.

Here is the list of requirements for this engine:

General Features		
Last Release	25/12/2007	✓
License	GPL	✓
Programming Language	Python	⚠
Data Storage	Files	✗
Page permissions	Yes	✓
Access control lists	Yes	✓
Authentication backend	LDAP	✓
Usability		
Preview	Yes	✓
Page history	Yes	✓
Revision differences	Between all	✓
Page index	Yes	✓
Interface languages	Over 40	✓
Page redirection	Yes	✓
Conflict handling	Conflict resolution	✓
Section editing	No	✓
Double click edit	Yes	✓
WYSIWYG editor	Yes	✓
HTML Tags	Some	✓
Access keys	No	✓
Output		
Printer Friendly	Print CSS	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Yes	✓
XML export	Yes	✓
PDF export	Plug-in	⚠
Media		
File attachments	Yes	✓
Embedded flash	Yes	✓
Embedded video	Yes	✓
Syntax	Different as the current syntax	⚠

Table 4: features of the MoinMoin engine

This engine was discarded for the same reasons as the TWiki one, explained before: it is written in the Python language (not supported by the platform) and its data storage is done by using files.

2.4.2. Engines which matched the minimum requirements

In this section we will find a list of engines which were studied carefully.

PmWiki

PmWiki is a wiki-based system for collaborative creation and maintenance of websites. PmWiki pages look and act like normal web pages, except they have an "Edit" link that makes it easy to modify existing pages and add new pages into the website, using basic editing rules. You do not need to know or use any HTML or CSS. Page editing can be left open to the public or restricted to small groups of authors.

A site administrator can quickly change the appearance and functions of a PmWiki site by using different skins and HTML templates. If you can not find an appropriate skin already made, you can easily modify one or create your own.

One principle of the PmWikiPhilosophy is to only include essential features in the core engine, but make it easy for administrators to customize and add new markup. Hundreds of features are already available by using extensions.

Here is the list of requirements for this engine:

General Features		
Last Release	22/01/2007	✓
License	GPL 2	✓
Programming Language	PHP	✓
Data Storage	Files, MySQL Plug-in	!
Page permissions	Yes	✓
Access control lists	Yes	✓
Authentication backend	LDAP, MySQL	✓
Usability		
Preview	Yes	✓
Page history	Yes	✓
Revision differences	Between all	✓
Page index	Yes	✓
Interface languages	Over 20	✓
Page redirection	Yes	✓
Conflict handling	Conflict resolution	✓

Section editing	Plug-in	!
Double click edit	Plug-in	✓
WYSIWYG editor	Plug-in	!
HTML Tags	Plug-in	!
Access keys	Yes	✓
Output		
Printer Friendly	Print View	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Plug-in	✓
XML export	Plug-in	✓
PDF export	Plug-in	!
Media		
File attachments	Optional	✓
Embedded flash	Plug-in	✓
Embedded video	Plug-in	✓
Syntax	Different as the current syntax	!

Table 5: features of the PmWiki engine

This engine brings many advantages. It can use a MySQL database to store its contents. It also permits authenticating users via an LDAP backend. It has an automated conflict resolution whenever two users try to edit the same page. A plug-ins system lets us load many features, such as a section editing system, a visual editor and an exporting tool.

Once studied carefully, we noticed this engine introduced a little problem: it was originally designed and developed by only one person: its creator (Patrick R. Michaud). This makes the code unintelligible; simply it cannot be understood by anyone but this person. The code does not seem to follow any structure, it is not commented and the documentation is conspicuous by its absence. This lack would make the integration of the engine a little bit more complicated than expected because we would have to spend lot of time by interpreting the original source code to be able to make the changes necessary to integrate it.

PhpWiki

PhpWiki is a web site where anyone can edit the pages through an HTML form. Linking is done automatically on the server side; all

pages are stored in a database. Installation of PhpWiki is as simple as uncompressing the source distribution.

Here's the list of requirements of this engine:

General Features		
Last Release	07/03/2006	✓
License	GPL	✓
Programming Language	PHP	✓
Data Storage	MySQL, etc.	✓
Page permissions	Yes	✓
Access control lists	Yes	✓
Authentication backend	DB, LDAP, etc.	✓
Usability		
Preview	Yes	✓
Page history	Yes	✓
Revision differences	Between all	✓
Page index	Yes	✓
Interface languages	English, Spanish	!
Page redirection	Yes	✓
Conflict handling	Conflict resolution	✓
Section editing	No	!
Double click edit	Optional	✓
WYSIWYG editor	Patch	!
HTML Tags	Some	✓
Access keys	No	✓
Output		
Printer Friendly	Print CSS	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Yes	✓
XML export	No	✓
PDF export	Yes	✓
Media		
File attachments	Yes	✓
Embedded flash	Yes	✓
Embedded video	Yes	✓
Syntax	Different as the current syntax	!

Table 6: features of the PhpWiki engine

This engine was released on its latest version 1.3 on March 7, 2006. It can use a MySQL database and can authenticate users via LDAP.

Allow users to compare different versions of a document. By installing a patch, it is possible to install a visual editor and the engine accepts some HTML tags inside the content.

This feature is directly related to one of the tasks of integrating the engine.

One of the objectives of the project is to let members of e-Catalunya to continue editing their pages created with the current wiki tool with the new one. We will refer to this process by the term "migration".

Migrating all of the existing pages in the current wiki to the newer one consists of translating their content from the syntax of the current engine to the one used by the new engine.

This "translation" is easier if the new engine accepts HTML, because by this we would only need to render each page and copy the resulting content to its corresponding page on the new engine.

This engine also has a built-in service which enables the final user to export pages to PDF.

When studying this engine we noticed a very big problem. Its latest version (1.3.14) was not stable. The community which developed this engine found a bug which made the engine to fall down due to unknown circumstances, or so they say.

This was unacceptable so we tried with the latest stable release of this engine, which was its version 1.2.10. We studied the features of this version of the engine and we missed many of the most important ones. In fact, the version 1.2.10 of this engine was so small that we considered it only as an editing and rendering pages tool, without, for instance, managing users. It was also very easy to install, so we tried it in the development environment. Soon we realized that such a simple engine would not lead us to anywhere.

TikiWiki

TikiWiki is an open source (LGPL) Content Management System (CMS) Groupware web application enabling websites and portals on the internet and on intranets and extranets. TikiWiki can be used as a structured wiki, a bug tracking system, a collaboration platform, a knowledge base, a blogging system or an Internet forum. TikiWiki is a customizable modular multi-feature package; each component can be enabled or disabled and customized by the TikiWiki administrator. TikiWiki extends the customization to the user with selectable skins and themes.

Here is the list of requirements of this engine:

General Features		
Last Release	26/10/2007	✔
License	LGPL	✔
Programming Language	PHP	✔
Data Storage	MySQL, etc.	✔
Page permissions	Yes	✔
Access control lists	Yes	✔
Authentication backend	LDAP, etc.	✔
Usability		
Preview	Yes	✔
Page history	Yes	✔
Revision differences	Between all	✔
Page index	Yes	✔
Interface languages	34	✔
Page redirection	Plug-in	✔
Conflict handling	Conflict Detection	✔
Section editing	Patch	!
Double click edit	Yes	✔
WYSIWYG editor	Patch	!
HTML Tags	Some	✔
Access keys	No	✔
Output		
Printer Friendly	Print View	✔
Themes & skins	Yes	✔
RSS feeds	Yes	✔
Auto-table-of-contents	Yes	✔
XML export	Yes	✔
PDF export	Yes	✔
Media		
File attachments	Yes	✔
Embedded flash	Yes	✔
Embedded video	Yes	✔
Syntax	Visual editor	✔

Table 7: features of the TikiWiki engine

The main characteristic of this engine is that it does not only provide a wiki engine, but offers many other features such as forums, blogs, articles, image and file galleries, maps, calendars, chats, etc. It is a platform similar to e-Catalunya. This is not really a big problem, but it would induce more time spent on integrating the engine, because we would have to disable and check those unwanted features. Another inconvenience of this kind of software is that usually the bigger they are, the slower they become.

Daisy

Daisy is a content management system that offers rich out-of-the-box functionality combined with solid foundations for extensibility and integration. Due to its genericity and flexibility, Daisy can be used for many different purposes, but is ideally suited for information-rich, structured content and asset management applications. Even for advanced content management applications, Daisy can be used and configured without any Java-coding skills: Daisy offers a Javascript/Cocoon-based extension framework.

Here is the list of requirements of this engine:

General Features		
Last Release	04/09/2007	✓
License	Apache License v2	⚠
Programming Language	Java	✓
Data Storage	Files, MySQL	✓
Page permissions	Yes	✓
Access control lists	Yes	✓
Authentication backend	NTLM, LDAP	✓
Usability		
Preview	Yes	✓
Page history	Yes	✓
Revision differences	Between all	✓
Page index	Yes	✓
Interface languages	English, Spanish	✓
Page redirection	No	✓
Conflict handling	Page Locking	✓
Section editing	Optional	✓
Double click edit	No	✓
WYSIWYG editor	Yes	✓

HTML Tags	All	✓
Access keys	No	✓
Output		
Printer Friendly	Print view	✓
Themes & skins	Yes	✓
RSS feeds	Yes	✓
Auto-table-of-contents	Yes	✓
XML export	Optional	✓
PDF export	Optional	✓
Media		
File attachments	Yes	✓
Embedded flash	Yes	✓
Embedded video	Optional	✓
Syntax	Visual editor	✓

Table 8: features of the Daisy engine

This engine is under the Apache License v2. Firstly we did not know much about this license, that's why we contacted lawyers to ensure its compatibility with the GPL license used in the platform. Meanwhile, we noticed this engine is also a kind of complete content management system. In the other side, it is coded in the Java programming language. As the rest of the platform is also coded in Java it would be easier to integrate it (we may not have to adapt the working environments to accept any new programming language) and it would be easier to understand due to our experience and skill in this technology.

Finally, we were reported that this license is not compatible at all with the GPL. This means that in order to integrate this engine we should meet certain conditions, the most important of which is that we should distribute the engine separately and we would not be able to modify directly the code of it. The only solution would be to extend the each code we need to modify. This is not very comfortable.

XWiki

XWiki is the current engine used for the wiki tool. This means that if this engine is finally selected we would have to talk about an upgrade rather than an integration of a new engine.

This engine offers both a generic platform for developing collaborative applications using the wiki paradigm and products developed on top of it. All XWiki software is developed in Java and under the LGPL open source license.

Here is the list of requirements of this engine:

General Features		
Last Release	25/06/2008	✔
License	LGPL	✔
Programming Language	Java	✔
Data Storage	MySQL, PostgreSQL, etc.	✔
Page permissions	Yes	✔
Access control lists	Yes	✔
Authentication backend	XWiki, LDAP, eXo Platform	✔
Usability		
Preview	Yes	✔
Page history	Yes	✔
Revision differences	Between all	✔
Page index	Yes	✔
Interface languages	English, Spanish, Catalan, etc.	✔
Page redirection	Yes	✔
Conflict handling	Page Locking	✔
Section editing	Yes	✔
Double click edit	No	✔
WYSIWYG editor	Yes	✔
HTML Tags	All	✔
Access keys	Yes	✔
Output		
Printer Friendly	Print view	✔
Themes & skins	Yes	✔
RSS feeds	Yes	✔
Auto-table-of-contents	Optional	✔
XML export	Yes	✔
PDF export	Yes	✔
Media		
File attachments	Yes	✔
Embedded flash	Yes	✔
Embedded video	Yes	✔
Syntax	The same	✔

Table 9: features of the XWiki engine

The problem of this engine is that we have used it for long time and our experience with it has not been much satisfactory. We know it is a heavy and slow engine even in its last release. Although it is already adapted to our environment and it's really just another possibility, we would not like to choose this way.

Specific e-Catalunya wiki engine

As the result of studying the available wiki engines we noticed that every single engine studied presented some problems.

This is the reason why we studied also the possibility of developing the new engine starting from scratch. It would be easier to integrate and it would fit better the user's needs, as it would be developed exclusively for them. Obviously, this option meets with all of the specified requirements. In the opposite, it would introduce programming errors which would have to be tested before releasing it.

The overall impression was that this option was the best one we could chose.

2.4.3. Decision

The final decision about which engine should be integrated was not on our hands. The direction of the e-Catalunya project falls on the *Generalitat de Catalunya*. This means that they had the last word on this decision.

That's why we put together all this information and we delivered a final report to this direction team, at *Generalitat de Catalunya* suggesting to develop the new wiki engine specifically for e-Catalunya. This final report was approved. The best option was to develop a brand new engine, starting from scratch. This new wiki engine, as following the politics of the platform, would be named: "**e-Wiki**".



3. Requirement analysis

This chapter focuses in the acquisition and description of system requirements. This will include all of the requirements for the new e-Wiki engine.

Software requirements are classified into 2 different families: functional requirements and non-functional requirements.

Functional requirements specify particular behaviours of a system and define functions of a software system. A function is described as a set of inputs, the behaviour, and its outputs. They are supported by non-functional requirements.

Non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviours. Non-functional requirements impose constraints on the design or implementation, such as performance requirements, accessibility, or usability.

3.1. Functional requirements

In this section we will see the functional requirements for the new e-Wiki. Basically we will see what members of e-Catalunya want the new e-Wiki for.

We will first see functional requirements of the new tool as a wiki engine and then we will see those related to this new e-Wiki as a new tool in the e-Catalunya platform: its integration.

3.1.1. E-Wiki features

As a new wiki engine, it is expected to have many implicit features which all wiki engines have. Let us specify them here.

As said before in section 2.1, a wiki is a collection of web pages designed to enable anyone who accesses it to contribute or modify content. This is the main concept. What we want is to adapt this main concept to the needs of e-Catalunya members.

By the feedback that we frequently have with some e-Catalunya users, we know that what they want is a kind of collaborative editing document manager. They want a tool which lets them create documents as they do with some office tools such as Microsoft Word or OpenOffice Writer in a collaborative scope. They do not want to spend time by learning new wiki syntax. They want an easy way to create and manage written documents.

E-Wiki: a wiki

A wiki tool in e-Catalunya will basically **consist of documents and its capability to manage them**. At present, users can only view a list of all of the documents contained in a wiki. This is a low level of management. In the new wiki engine we will let them not only list all of the documents in many different ways, but also remove documents to a trash bin, restore or delete them or change its identifier, if needed.

A wiki in e-Catalunya, as all the other tools, will be able to be opened or closed. This is directly related to the group in which is placed the tool. When a group is closed, all of its tools are closed too. The closed state prevents any interaction with the tool. A closed group can only be accessed for reading its content, but not for writing it.

Structure of a document

Documents in e-Wiki will consist of two types of data:

- Its **content**: this will be the body of the document.
- Some metadata: this will consist of some useful information such as the **language**, in which the document is written, its **identifier**, its **title**, and some **related words** (also known as key words).

By taking a look at these meta-data, we notice some things. The first one is that there appears the "**language**" property of a document. This property deals with the multilingual capability of the entire e-Catalunya platform. e-Catalunya is a multilingual web platform that enables members to see the platform in three different languages: Catalan, Spanish and English. This multilingual capability, usually, only affects to the structure of e-Catalunya, but not its contents, because many type of contents in e-Catalunya do not have this *language* property.

So members will be able to write documents in the three languages defined in the platform. Obviously, translations of documents will rest in the hands of members as there will not be any automated translating tool in e-Wiki. The behaviour of showing one translation or another will be the same as the rest of the platform. If a Spanish member accesses a document which is written in English and Spanish, the platform will firstly show him the Spanish version and will offer him the possibility of view it in English.

The second important metadata is the **identifier**. Identifiers of documents will be unique in a single wiki tool and they will identify a single document written in a concrete language. This will be used to create internal links between documents in a single wiki. The meaning of two documents having the same identifier is that they are translations one of each other. It is mandatory for two documents

having the same identifier to be written in different languages. To ease things, we only will enable the following basic characters and numbers.

- From "A" to "Z"
- From "a" to "z"
- From 0 to 9

By only enabling these characters in the identifier we prevent many problems such as misspelling. For instance, English keyboards do not have tildes in their vocals, so it will not be easy to search for a document if this contains tildes in its identifier.

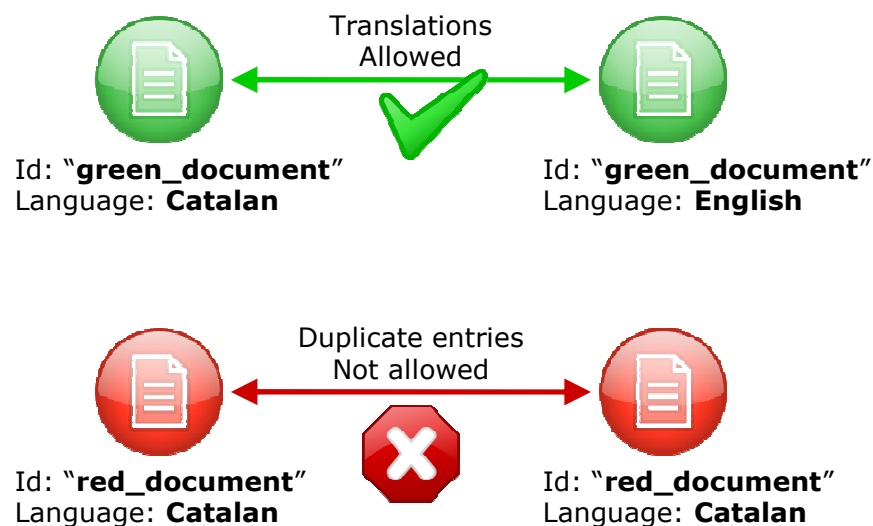


Figure 16: identifier and language properties of a document

The purpose of the **title** is to enable the use of special characters and spaces to graphically identify a document. In addition, as any restriction won't affect this attribute, different translations of the same document will be able to have their own title. This will be useful because it is expected for the title to be written in the same language as the document is.

Viewing a document

Documents will be stored in HTML language. This means that the visualization of it only involves rendering the document directly on the browser the user is using.

We want to include some features in our wiki engine, as many other wiki engines do. One of them is to **keep a history of each contained document** by saving a copy of it at any time it is edited and saved. With this history we can offer the possibility of surfing across all of the versions of a document giving a general overview of how the document is evolving. This feature opens the possibility of offering to the user the capability of easily **compare two versions** and to recover an older version of the document if desired.

A part from that, it will be useful for e-Catalunya members to let them **export documents to different formats**, such as PDF (portable document format) or ODF (open document format).

Editing a document

Edit documents is the principal action users have to do in order to create and give sense to their wikis. But this kind of interaction implies some problems. The most important one is how to manage concurrency.

A concurrency conflict appears in a wiki when two different users edit the same document at the same time. Both users start to edit the document at the same state. The first user who saves the document will lose his changes when the second user saves the document because this second user does not know about the changes done by the first one. There exist some mechanisms that avoid this problem.

There are many ways to avoid concurrency. Some of them try to merge the changes done at the same time in a document, but it may not be as reliable as desired. The simplest way to avoid

concurrency is to avoid two users editing the same document at the same time. To do this, any time a user wants to edit a document we will give him the control over the document for a lease time by locking the document. Before this lease time expires, the user should apply the changes and save the document. If any other user tries to edit a locked document the system will not let him do it and will give him some information about who is editing it and how much time remains until the document gets available again. As we think this may induce some trouble, we will let users the possibility of manually unlock the documents, warning them about consequences.

This mechanism solves the concurrency problem, but adds another one: the document may not be editable for a long time. To smooth a little this problem we will let users to edit only little sections of the entire document.

So, finally, documents will be composed of ordered sections, and sections will store the content of the document. Sections will also store some information about who edited it for the last time and when did it happen.

WYSIWYG Editor

As said before, we do not want users to spend time learning wiki syntax different from the used in the current wiki tool. To avoid this problem we can maintain the same syntax or simply stop using it.

The second option is more attractive because users who did not know this old syntax will be able to use this new tool without the lack of having to learn special syntax.

To enable this feature we will use a visual editor which is still used in other parts of the platform. This editor is an open source tool called FCKEditor³.

Document lifecycle

So members are expected to create documents and edit them at will. But documents can also be removed to a trash bin. The following diagram shows the different states a document can have:

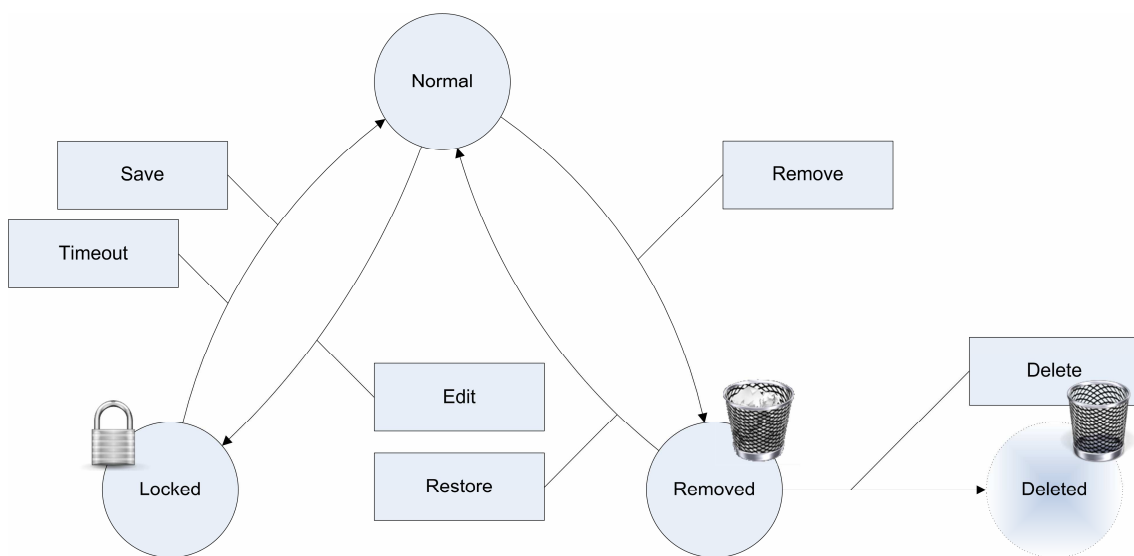


Figure 17: document state diagram

A document can be **edited or removed**. Once is being edited, it steps to the “blocked” state. In this state it can not be removed, it can only step back to the “normal” state by being saved or by exceeding the timeout (set to 30 minutes). If a document is removed, then it steps to the “removed” state. In this state can not be edited, it can only step back to the “normal” state by restoring it or step *beyond* to the “deleted” state, which basically consist of completely disappearing from database.

³ You can try it and download it at <http://www.fckeditor.net/>

E-Wiki: comments

As happens in other tools in the platform, the content of the e-Wiki tool is suitable of being commented by other users. We will ease the way of making comments to a document and managing them.

3.1.2. E-Wiki tool in e-Catalunya

Now we are going to see what functional requirements we do need for this engine to be able to integrate it totally with the e-Catalunya platform.

As other tools in e-Catalunya, the new e-Wiki has a life cycle. This life cycle is composed of very few steps.

The lifecycle of a tool in e-Catalunya starts when it is assigned to any of the available entities in e-Catalunya (a portal, a group or a member). In our particular case, the new e-Wiki will be able to be assigned only to portals and groups. We do not want to let members have their own tools, for the moment, because it seems useless, as the purpose of wikis is to let many users contribute, not only one.

At the time an administrator assigns a tool to an entity, he has to set its permissions. This permissions will determine which members will be able to interact with the tool and which level of interaction will they have.

Permissions

Each tool has different permissions. Each one of these permissions represents some basic actions available with the tool.

In our case, we will set four different permissions:

- View documents permission
- Edit documents permission
- Manage documents permission
- Make comments permission

By taking a look at the names we can get a general idea about what these permissions will let users do and what will not let users do. Let us detail them. In the following table, each column represents one permission and each row represents one action available.

<u>Actions</u>	<u>Permissions</u>			
	<u>View</u>	<u>Edit</u>	<u>Manage</u>	<u>Comment</u>
View a document	✓			
Export a document	✓			
View the history	✓			
View a old version	✓			
Create a document		✓		
Edit a document		✓		
Lock a document		✓		
Revert a document		✓		
Clear the history			✓	
Unlock a document			✓	
Rename a document			✓	
Remove a document			✓	
Restore a document			✓	
Delete a document			✓	
Comment a document				✓
Edit a comment				✓ ⚠
Delete a comment				✓ ⚠

⚠: A user can only **edit** or **delete comments** if he is a moderator **or** he has this permission **and** he is the owner of the comment.

Table 10: relation of permissions and its affected actions

By that, if a member of e-Catalunya has the permission of viewing documents, he will be able to view every document contained in the specified wiki, export it, access its history and also see the content of older versions of it.

Each one of these permissions will have to be matched with any of the following roles of the e-Catalunya platform (sorted from the most restrictive to the less one):

- Group administrators
- Group moderators
- Group members
- Portal members

- Guests

These roles are cumulative. This means that each role includes the ones above it.

*For instance, if the "**view permission**" is set to **group members**, then **group moderators** and **group administrators** will also have this permission, but **portal members** and **guests** will not have it.*

We should be careful about one thing with these permissions. There's the possibility of setting permissions which may not be consistent. What would happen if the permissions of a wiki let a member of e-Catalunya edit a document but does not let him view it? How a member of e-Catalunya it's supposed to edit a document if he can not see it, and even does not know about it?

The solution of this problem relies on the fact that only group administrators are able to assign and delete tools. As they are the only ones that can do this, they will have the responsibility of setting consistent permissions. Fortunately, if they fail on setting the permissions when creating a wiki tool, they will be able to change the permissions at any time through the administration page.

3.2. Non-functional requirements

In this section we are going to see non functional requirements for the new wiki tool. As said before, non functional requirements are constraints and behaviours the wiki engine must fulfil.

3.2.1. e-Catalunya platform independent modelling

This title may cause a little confusion. We do not want the new e-Wiki to be completely independent from the rest of the platform, of course. This has to do with the open source property of the platform. As this platform is open source, everyone is able to take its source code and use it at will. As we are developing a new wiki engine, we think it may be useful for other developers all over the world to design the engine of this tool as modular as possible. We want to develop a wiki engine with a low level of cohesion.

Modular programming is a software design technique that increases the extent to which software is composed from separate parts, called modules. Conceptually, modules represent a separation of concerns, and improve maintainability by enforcing logical boundaries between components.

Cohesion is a measure of how strongly-related and focused the various responsibilities of a software module are. Cohesion is an ordinal type of measurement and is usually expressed as "high cohesion" or "low cohesion" when being discussed.

This means that the modelling of the engine has to be independent from the platform, and then we will have to integrate this model to the platform needs.

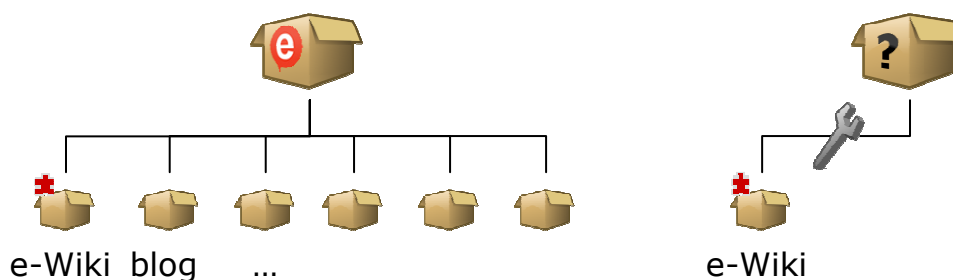


Figure 18: our wiki engine may be useful for other platforms.

3.2.2. Visual integration

System integration is the bringing together of the component subsystems into one system and ensuring that the subsystems function together as a system.

Visual integration has to do with the interface of the tool. The final result from the point of view of a member of e-Catalunya is a set of web interfaces that will let him use the wiki. These web interfaces have to follow the same structure as other tools in e-Catalunya, as they are part of one of these tools. Final users are accustomed to the current interface.

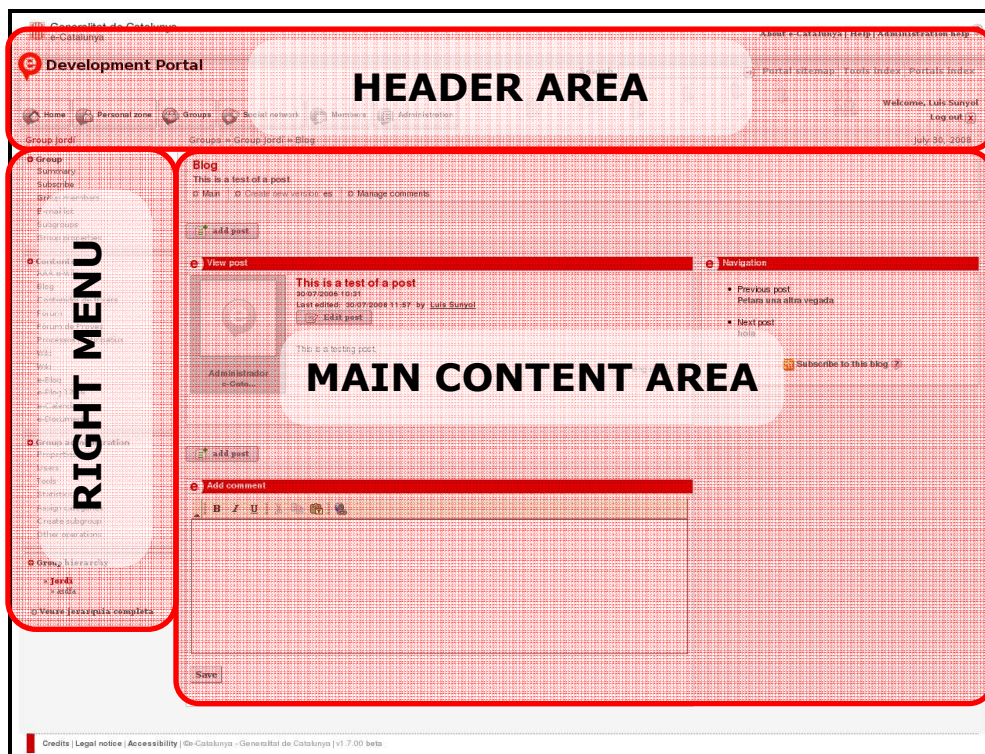


Figure 19: visual integration

Visual integration deals with structure, content, behaviour, etc. This means that we will have to maintain not only the structure, but also the aspect. This deals with colours, font types and sizes, buttons, links, images, etc.

3.2.3. Accessibility

Accessibility is a general term used to describe the degree to which a product is accessible by as many people as possible. Accessibility is often used to focus on people with disabilities and their right of access to entities, often through use of assistive technology.

Our purpose is to design the new e-Wiki the most accessible as possible. As the rest of the platform is compliant with the WAI's "AA" accessibility level, the e-Wiki will not be different.

WAI is the acronym of Web Accessibility Initiative. It works with organizations around the world to develop strategies, guidelines, and resources to help make the Web accessible to people with disabilities.

Accessibility involves not only people with disabilities, but also takes care about technologies. For instance, an internet browser installed on handheld devices, which work on batteries, may not have the same availability of CPU as a browser installed on a personal computer. This may cause a lack of technologies in these browsers, such as scripting languages (for instance, JavaScript). We must take care about this, and make every browser to be able to offer the same functionalities to the final user.

3.2.4. Usability

Usability is a term used to denote the ease with which people can employ a particular tool or other human-made object in order to achieve a particular goal. Usability refers to the elegance and clarity with which the interaction with the tool is designed.

Our purpose is to design the new e-Wiki the most usable as possible. We will achieve that by contacting some experts on this field which will give us some advices.

For instance, let us suppose that someone is going to make an irreversible action, such as deleting a document, we would be improving its usability if we make the system to ask him for a confirmation before deleting it definitely. The reason to act like this is that the user may have clicked on the delete button it by error.



Figure 20: Usability improvement

3.2.5. Reliability

Reliability is the ability of a system to perform and maintain its functions in routine circumstances, as well as hostile or unexpected circumstances.

We do not want the user to wait for a long time to get a requested document. We want to speed up the interaction as much as possible. We want to shorten the time between a user requests a document and the system serves it as much as possible. If we do not care that, it will not matter how good the wiki engine is, thus no one will use it, as waiting for anything is one of the most detestable things you can expect from a web page. Slow systems are sentenced to fail.

We must take care that the community is currently growing faster than ever and there are over 10.000 members in the platform. We must ensure the engine will be able to bear the entire load this may produce.

It is said that you have a matter of seconds before you start losing readers, when loading your web page.

3.2.6. Security

All of the personal information managed by this engine must be stored and used in an LOPD (*Ley Orgánica de Protección de Datos*) compliant way. LOPD is the Spanish law which's object is to grant and protect personal data from individuals.

Other data, such as the content of documents, has to be filtered in order to ensure the proper running of the platform. By filtering this code we prevent malicious code from being published. Also filtering has to do with correctness. Correctness is needed to compile with the accessibility requirement in the way that content set by members is later available to general public. These members don't need to know about accessibility, so we must take care of it for them. These filters are already developed and fully tested. They only have to be integrated.

The semantic of the content is directly managed by the moderators of the platform. They are responsible for preventing any kind of unwanted content from being published.



4. Specification

This chapter will focus on system specifications.

A specification is a type of a standard which provides the necessary details about the specific requirements.

First of all we are going to see which actors are involved. Then we will see which use cases will be able to use these actors.

4.1. Actors

First of all, let us see what an actor is. An actor is something or someone who supplies a stimulus to the system. An actor cannot be controlled by the system and is defined as being outside the system. An actor is often thought of as a role, rather than an actual person. A single person in the real world can be represented by several actors if they have several different roles and goals in regard to a system.

Every actor appearing here in this chapter can represent many members of e-Catalunya, depending on their roles and the scope where the tool is placed.

As said in section 3.1, there are four permissions that need to be set up when assigning a wiki tool to a group or a portal. These permissions will enable or disable some actions to different roles on the group or portal in which the tool is placed.

In this specification, we will have 3 basic actors:



Reader: someone who may not be directly related to the group or the portal in which the wiki is placed. This actor will only have permission to view documents and make comments.



Writer: someone who may be directly or indirectly related to the group or the portal in where the wiki is placed. This actor will have permissions to view, create and edit documents, and also will be able to make comments as well as reader actor can.



Moderator: someone who may be directly related to the group or the portal where the wiki is placed. This actor will have all of the permissions of the tool. A moderator will be able not also to view, create and edit documents,

but also to remove, recover, delete, etc. them in the wiki tool.

Summarizing:




Actors	Permissions			
	View	Edit	Manage	Comments
 Reader	✓	✗	✗	✗
 Writer	✓	✓	✗	✗
 Moderator	✓	✓	✓	✓

Table 11: table of permissions for each actor

Apart from that, we will use also some roles of the platform, described in section 1.3.3, as actors. It must be said that these actors do not correspond to any of the roles of the platform. These actors can be played by all of the available roles of the platform by correctly setting the permissions of the tool. We use them only to ease the specification task.

For instance, we can create a public wiki tool inside a public portal, and set its four permissions to the public members. By this anyone, even anonymous users, will be able to view, create, remove, etc. documents.

In the other hand, we can create a wiki tool in a private group and set its four permissions only to administrators of that group. By this, only administrators of the group will be able to access the wiki and contribute with their own.

4.2. Use cases

A use case is a description of a system's behaviour as it responds to a request that originates from outside of that system. Use cases describe the interaction between a primary actor (the initiator of the interaction) and the system itself, represented as a sequence of simple steps.

In the e-Catalunya platform there are some generic use cases for every tool, such as assigning or deleting tools from groups or portals. These use cases must be taken into account also when designing them all.

Next we will see all of the use cases available in the final e-Wiki.

4.2.1. Assign a new e-Wiki to a group or a portal

Brief description

This is the first use case available in the platform in order to use the e-Wiki tool.

Actors

Administrator of a group or a portal

Basic flow of events

The use case starts when a group or a portal administrator selects the new e-Wiki tool in the group or portal administration tools page and clicks on the "Assign tool" button.



(Administrator of a group or a portal)



(System)

1. The administrator clicks on the assign tool button with the e-Wiki combo box selected.

2. The system asks for the desired name of the new e-Wiki, its accessibility and its permissions.

3. The administrator selects the correct configuration and clicks on assign tool.

4. The system assigns the tool to the specified group or portal with the specified configuration.

Alternate flow: cancelling



1. The administrator clicks on the assign tool button with the e-Wiki combo box selected.

2. The system asks for the desired name of the new e-Wiki, its accessibility and its permissions.

3. The administrator cancels the use case. The use case ends here.

Alternate flow: unable to create a wiki



1. The administrator clicks on the assign tool button with the e-Wiki combo box selected.

2. The system asks for the desired name of the new e-Wiki, its accessibility and its permissions.

3. The administrator selects the correct configuration and clicks on assign tool.

4. The system can not create the e-Wiki because of any possible problem. The system warns the user about the problem and the use case ends up here without creating the new e-Wiki.

4.2.2. Close an e-Wiki

Brief description

This use case is another inheritance from the e-Catalunya platform. Each tool has to be able to get closed. This means that the content will remain in a read-only state.

Actors

Administrator of a group or a portal

Basic flow of events

The use case is called when a group or portal administrator wants to close its group or portal.



(Administrator of a group or a portal)



(System)

1. The administrator closes its group or portal.

2. The system asks for a confirmation.

3. The administrator confirms the action.

4. The system closes the group or portal and all its assigned tools.

Alternate flow: cancelling



1. The administrator closes its group or portal.

2. The system asks for a confirmation.

3. The administrator cancels the use case.

4.2.3. Modify the properties of an e-Wiki tool

Brief description

This use case is used to change the configuration of an assigned e-Wiki.

Actors

Administrator of a group or a portal

Basic flow of events

The use case starts when an administrator of a portal or a group wants to edit the configuration of an assigned e-Wiki.



(Administrator of a group or a portal)



(System)

1. The administrator clicks on the edit e-Wiki link.

2. The system asks for the new desired configuration of the e-Wiki.

3. The administrator selects the correct configuration and clicks on the save button.

4. The system changes the configuration of the e-Wiki tool.

Alternate flow: cancelling



1. The administrator clicks on the edit e-Wiki link.

2. The system asks for the new desired configuration of the e-Wiki.

Step 3: The administrator cancels the use case.

4.2.4. Delete an e-Wiki tool

Brief description

This use case is used in order to delete an unwanted e-Wiki tool from a group or a portal.

Actors

Administrator of a group or a portal

Basic flow of events

The use case starts when the administrator selects the e-Wiki and clicks on the delete tool button in the tools administration page.



(Administrator of a group or a portal)



(System)

1. The administrator clicks on the delete button.

2. The system asks for a confirmation.

3. The administrator confirms the operation.

4. The system deletes the desired e-Wiki from its group or portal.

Alternate flow: operation not confirmed



1. The administrator clicks on the delete button.

2. The system asks for a confirmation.

3. The administrator does not confirm the operation. The use case ends here.

4.2.5. View a document

Brief description

This use case is the main use case available in the tool. It is used every time someone wants to read a document.

Actors

Reader, writer, moderator

Basic flow of events

In the basic flow of events the document object does exist and it is not removed.



(reader)



(System)

1. The reader or writer clicks on a link pointing at a written document of an e-Wiki.

2. The system returns the requested document.

Alternate flow: reader accessing a nonexistent document



(reader)



(System)

1. The reader clicks on a link pointing at a nonexistent document inside an e-Wiki.

2. The system can not find the requested document. The system warns the reader about the problem.

Alternate flow: writer accessing a nonexistent document



(writer)



(System)

1. The writer clicks on a link pointing at a nonexistent document inside an e-Wiki.

2. The system can not find the requested document. The system warns the writer about the problem and offers the possibility to rapidly create the requested document.

Alternate flow: reader or writer accessing a removed document

In this use case, the document object has been previously removed (see the use case "Remove document", specified in page 97).



(reader)



(System)

1. The writer clicks on a link pointing at a removed document of an e-Wiki.

2. The system tells the user that the document is removed.

Alternate flow: reader or writer accessing a removed document

In this use case, the document object has been previously removed (see the use case "Remove document", specified in page 97).



(moderator)



(System)

1. The moderator clicks on a link pointing at a removed document of an e-Wiki.

2. The system tells the user that the document is removed and offers the possibility of view it anyway.

3. The moderator clicks on the offered link to view the document.

4. The system returns the requested document.

4.2.6. Export a document

Brief description

This use case is used to export a document to a HTML format.

Actors

Reader

Basic flow of events

The use case starts from a document page.



(reader)



(System)

1. The reader clicks on the export link of a document.

2. The system returns the requested document formatted in an HTML downloadable file.

Preconditions

The document does exist and is not removed.

4.2.7. View the history of a document

Brief description

This use case gives information about how many versions of a document there are and what changes have been done between them.

Actors

Reader

Basic flow of events

The use case starts from a document page.



(reader)



(System)

1. The reader clicks on the history link.

2. The system returns the requested list of versions of the document.

Preconditions

The document does exist and is not removed.

4.2.8. View an old version of a document

Brief description

This use case is used when someone wants to view an older version of a document different from the last one.

Actors

Reader

Basic flow of events

The use case usually starts from the history page of a document.



(reader)



(System)

1. The reader clicks on a link to an older version of a document.

2. The system returns the requested version of the document.

Alternate flow: nonexistent version



1. The reader clicks on a link to an older version of a document.

2. The system can not find the requested version of the document. Warns the user about the problem.

Preconditions

The document exists and is not removed.

4.2.9. Create a document

Brief description

This use case is used to create a new document.

Actors

Writer

Basic flow of events



(writer)



(System)

1. The writer clicks on the "new document" button.

2. The system asks for some data such as the title and the content of the document.

3. The writer enters all the required data and clicks on "Save" button.

4. The system saves the new document to the database and gives a feedback message to the user.

Alternate flow: cancelling



1. The writer clicks on the "new document" button.

2. The system asks for some data such as the title and the content of the document.

3. The writer cancels the use case.

Alternate flow: wrong input data



1. The writer clicks on the "new document" button.

2. The system asks for some data such as the title and the content of the document.

3. The writer enters all the required data and clicks on "Save" button.

4. The system can not create the document because there is a lack on the input data, or there is another document with the same name and language in the wiki. The system jumps to step 2 maintaining all the information introduced in step 3 by the writer and warns him about the specific problem.

Preconditions

The wiki in where the document is created must not be closed.

4.2.10. Edit a document

Brief description

This use case is used to edit an already created wiki document. This use case only enables editing the document properties and content, but not its identifier and language.

Actors

Writer

Basic flow of events



(writer)



(System)

1. The writer clicks on the edit button when viewing a document.

2. The system locks the document and asks the writer for the new data.

3. The writer applies the changes he has in mind and clicks on the "Save" button before his lease time expires.

4. The system saves the document, unlocks it and gives a feedback message to the user.

Alternate flow: cancelling



1. The writer clicks on the edit button when viewing a document.

2. The system locks the document and asks the writer for the new data.

3. The writer clicks on the cancel button.

4. The system discards the changes done to the document and unlocks the document.

Alternate flow: exiting 1



1. The writer clicks on the edit button when viewing a document.

2. The system locks the document and asks the writer for the new data.

3. The writer exits the use case clicking wherever but on the cancel button.

4. The system warns the user about the consequences of this and asks him for a confirmation.

5. The user confirms the action.

6. The system discards the changes done to the document but does not unlock it.

Alternate flow: exiting 2



1. The writer clicks on the edit button when viewing a document.

2. The system locks the document and asks the writer for the new data.

3. The writer exits the use case clicking wherever but on the cancel button.

4. The system warns the user about the consequences of this and asks him for a confirmation.

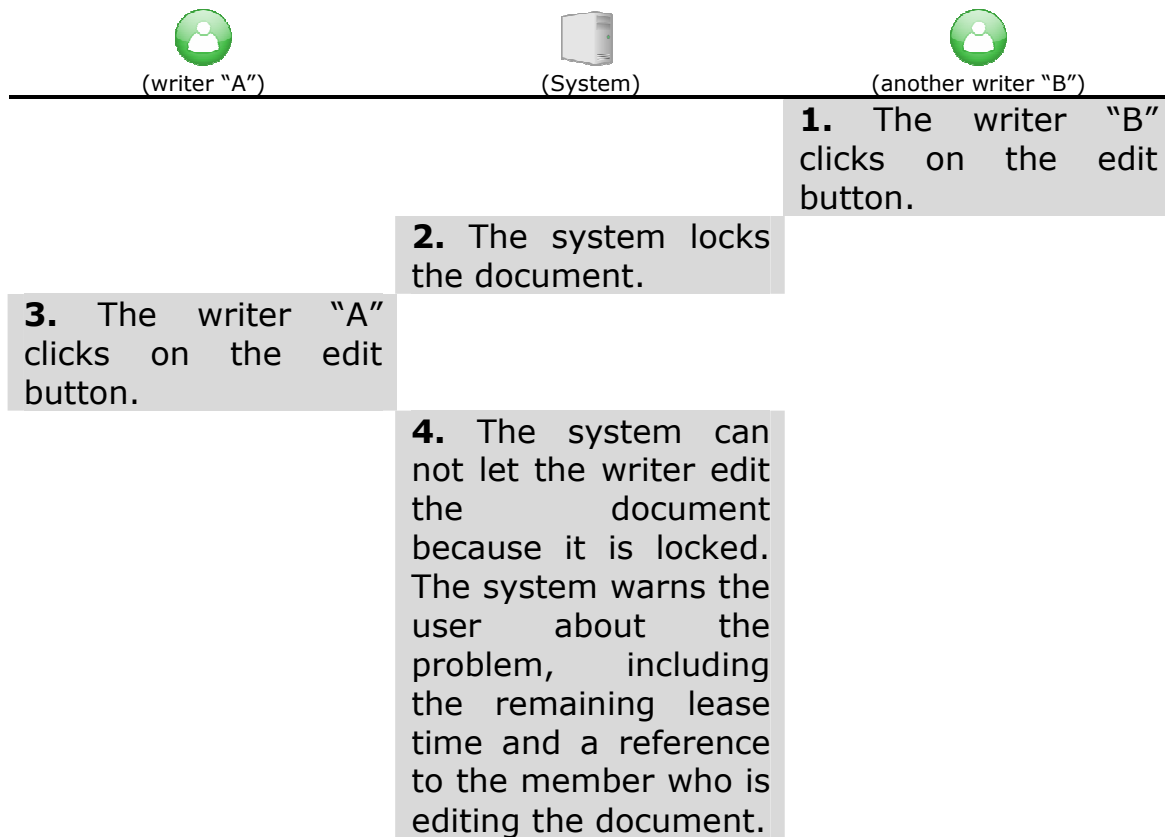
5. The user does not confirm the action.

6. The system jumps back to step 2, maintaining all the changes done by the writer to the document in step 3 but without saving them to database.

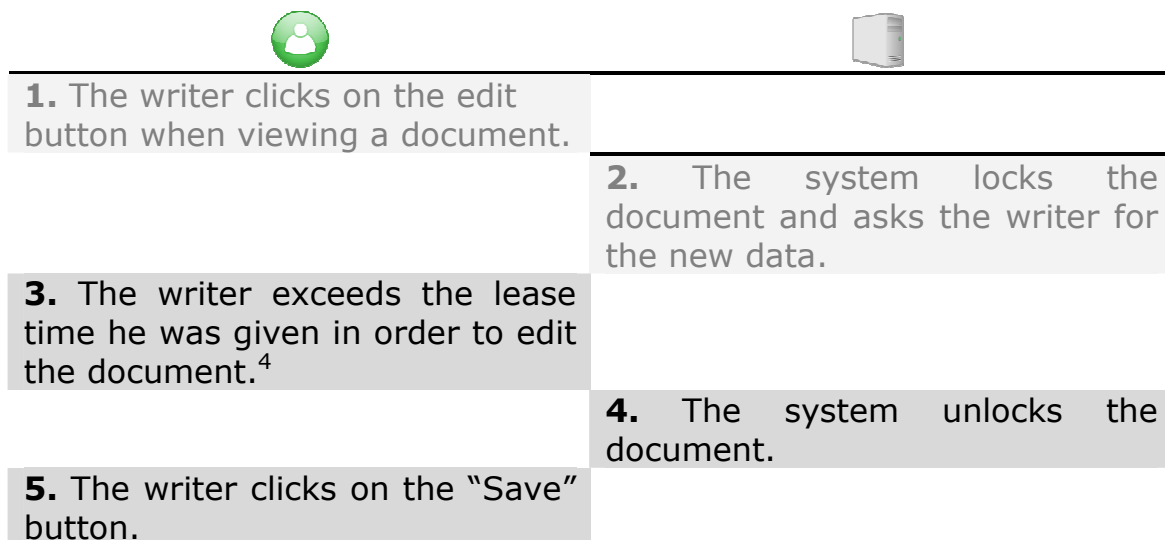
Alternate flow: the document is already being edited

In this flow of events appear two different writers. The first one (called "A") is the one that matters. The writer "A" is the one that

experiences this alternate flow of events. The other writer (called "B") is just there to ease the comprehension of the flow.



Alternate flow: lease time expired 1 (the "lucky" writer)

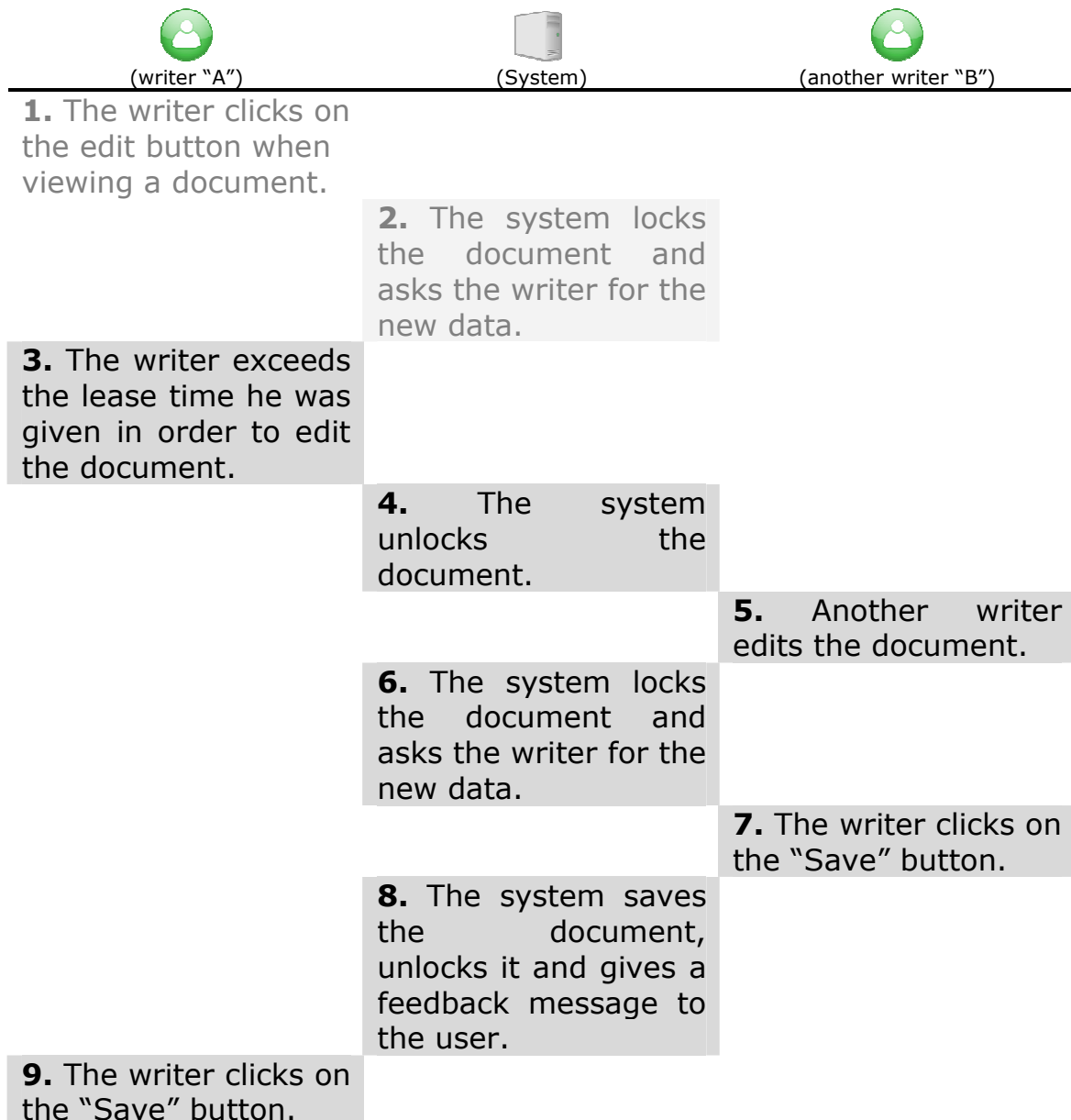


⁴ This step can be replaced by a moderator unlocking manually the document. This use case is shown later, on section 4.2.14. By one way or the other, the result is exactly the same: the document gets unlocked.

6. The system saves the changes like if nothing had happened and gives a feedback message to the user.

Alternate flow: lease time expired 2 (the "not-so-lucky" writer)

As before, this flow of events also needs the appearance of two different writers. Also as before, the one that matters is the first one, called "A".



10: As someone edited the document, the system warns the writer about the problem and jumps back to step 2, applying the changes done by the user in step 3 but without saving the changes to database.

Preconditions

The document does exist and it is not removed.

The wiki in where the document is placed must not be closed.

4.2.11. Upload a file

Brief description

This use case is used when a writer is editing a document and wants to attach a file or an image in it.

Actors

Writer

Basic flow of events

This use case can be used only when editing a document, specifically in the step 3 of the basic flow of events of the use case "Edit a document", just above this one.



(writer)



(System)

1. The writer clicks on the "Attach file" button.

2. The system opens a new window asking for the file and some properties such as legal terms.

3. The writer clicks on the "Upload" button.

4. The system attaches the file to the document.

Alternate flow: cancelling



1. The writer clicks on the "Attach file" button.

2. The system opens a new window asking for the file and some properties such as legal terms.

3. The writer cancels the use case by closing the new window.

Preconditions

The document exists, it is not removed and it is being edited by the same user which is attaching a file.

The wiki in where the document is placed must not be closed.

4.2.12. Revert a document

Brief description

This use case is used when someone wants to restore an old version of a document.

Actors

Writer

Basic flow of events

The use case starts from the history page of a document.



(writer)



(System)

1. The writer selects one of the old versions listed and clicks on the "Revert" button.

2. The system retrieves the document to the specified version and gives a feedback message to the user.

Preconditions

The document exists and it is unlocked.

The wiki in where the document is placed must not be closed.

4.2.13. Clear the history of a document

Brief description

This use case is used to clear the history of a document. The history of the document will be cleared and left only the latest version.

Actors

Moderator

Basic flow of events

The use case starts from the listing versions page.



(moderator)



(System)

1. The moderator clicks on the "Clear history" button.

2. The system clears the history of the document and gives a feedback message to the user.

Preconditions

The document exists and it is unlocked.

The wiki in where the document is placed must not be closed.

4.2.14. Unlock documents

Brief description

This use case is used to manually unlock a set of previously locked documents. It would be desirable not to enable this feature, because documents get automatically unlocked when the lease time expires, but we know that sooner or later we will be asked to implement this feature.

Actors

Moderator

Basic flow of events

The use case starts from the page where all the existing documents of the wiki are listed.



(moderator)



(System)

1. The moderator of the tool selects as many documents as desired and clicks on the "Unlock" button.

2. The system forces the unlocking of all of the documents and gives a feedback message to the user.

Preconditions

All of the documents selected must exist. The documents are expected to be previously locked, but it is not mandatory. If any of the documents selected to be unlocked is not locked, then the system does nothing.

4.2.15. Rename documents

Brief description

This use case is used to change the identifier of documents. Changing the identifier of a document will make all of the links pointing at it to be outdated. This is not desirable, and that is why this use case is expected to be used when we know the document is not linked in any other document. Otherwise, it would be necessary to review and edit those documents which contain links pointing at this one.

It should be said that changing the identifier of a document implies changing the identifier of all of its translations.

Actors

Moderator

Basic flow of events

The use case starts from the page where all of the documents of the wiki are listed.



(moderator)



(System)

1. The user selects the documents he wants to change its identifier and clicks on the "Change Id" button.

2. The system asks the moderator for the new identifiers for the documents

3. The moderator sets the new names for the selected documents and clicks again on the "Change Id" button.

4. The system changes the names of the documents and gives a feedback message to the user.

Alternate flow: cancelling



1. The user selects the documents he wants to change its identifier and clicks on the "Change Id" button.

2. The system asks the moderator for the new identifiers for the documents

3. The moderator cancels the use case by clicking anywhere but the "Change Id" button.

Preconditions

All of the selected documents exist and they are not locked by anyone.

The wiki in where the documents are placed must not be closed.

4.2.16. Remove documents

Brief description

This use case is used to remove documents. In fact, this use case moves the documents to a trash bin and they remain in database. Removed documents are expected to be restored or deleted, but not to remain in this state for much longer.

Removed documents will be only accessible for moderators.

Actors

Moderator

Basic flow of events

The use case starts from listing documents page.



(moderator)



(System)

1. The moderator selects the documents he wants to remove and clicks on the "Remove" button.

2. The system searches for translations of each selected document and offers to the moderator the possibility to also remove them.

3. The moderator chooses which translations will be removed and which will not and clicks on the "Remove" button.

4. The system asks the user for a confirmation.

5. The moderator confirms the action.

6. The system removes the selected documents and translations and gives a feedback message to the user.

Alternate flow: cancelling



1. The moderator selects the documents he wants to remove and clicks on the "Remove" button.

2. The system searches for translations of each selected document and offers to the moderator the possibility to also remove them.

3. The moderator cancels the use case by clicking anywhere but the "Remove" button or chooses which translations will be removed and which will not and clicks on the "Remove" button.

4. The system asks the user for a confirmation.

5. The moderator cancels the use case by clicking anywhere but the "Remove" button.

Preconditions

All of the documents exist and are not locked by anyone.

The wiki in where the documents are placed must not be closed.

4.2.17. Restore documents

Brief description

This use case is used to restore documents previously removed. This is one of the two options available on removed documents.

Actors

Moderator

Basic flow of events



(moderator)



(System)

1. The moderator selects the documents to restore and clicks on the "Restore" button.

2. The system asks the user for a confirmation.

3. The moderator confirms the action.

4. The system restores the selected documents and gives a feedback message to the user.

Alternate flow: cancelling or not confirming



1. The moderator selects the documents to restore and clicks on the "Restore" button.

2. The system asks the user for a confirmation.

3. The moderator cancels the use case by clicking on the cancel button or exiting.

Preconditions

The selected documents do exist and they have been previously removed.

The wiki in where they are placed has not been closed.

4.2.18. Delete documents

Brief description

This use case is used to delete documents previously removed. This is the second option available on removed documents.

Actors

Moderator

Basic flow of events



(moderator)



(System)

1. The moderator selects the documents he wants to delete and clicks on the "Delete" button.

2. The system asks the user for a confirmation.

3. The moderator confirms the action.

4. The system deletes the selected documents and gives a feedback message to the user.

Alternate flow: cancelling or not confirming



1. The moderator selects the documents he wants to restore and clicks on the "Delete" button.

2. The system asks the user for a confirmation.

3. The moderator cancels the use case by clicking on the cancel button or exiting.

Preconditions

The selected documents do exist and they have been previously removed.

The wiki in where they are placed has not been closed.

4.2.19. Comment a document

Brief description

This use case is used to make comments to a document.

Actors

Reader

Basic flow of events

The use case starts from the home page of a document.



(reader)



(System)

1. The reader writes a comment in the comment box and clicks on the "Save" button just below it.

2. The system saves the comment.

Alternate flow: anonymous reader makes a comment

When the user who is making a comment is anonymous he will have to fulfil a CAPTCHA. This measure is done to avoid spam coming from web robots, specifically forum spam bots.



(anonymous reader)



1. The reader writes a comment in the comment box, fulfils the CAPTCHA box and clicks on the "Save" button just below it.

2. The system checks the CAPTCHA is correctly answered and saves the comment.

A CAPTCHA is a type of challenge-response test used in computing to ensure that the response is not generated by a computer, in our case a web robot. The process usually

involves one computer (a server) asking a user to complete a simple test which the computer is able to generate and grade. Because other computers are unable to solve the CAPTCHA, any user entering a correct solution is presumed to be human.

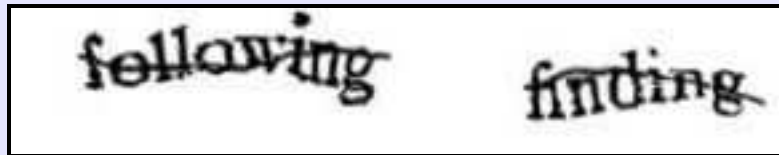


Figure 21: An example of a CAPTCHA challenge, containing the words "following finding".

Forum spam bots are a kind of web robots. Web robots, also known as bots, are software applications that run automated tasks over the Internet. Forum spam bots surf the web, looking for guestbooks, wikis, blogs, forums and any other web forms to submit spam links to the web forms it finds.

Preconditions

The document exists and it is not removed.

The wiki in where they are placed has not been closed.

4.2.20. Edit a comment

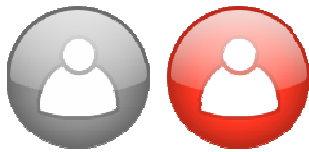
Brief description

This use case is used to edit a comment of a document.

Actors

Author of the comment or Moderator of the group or portal.

Basic flow of events



(author, moderator of the group/portal)



(System)

1. The user clicks on the "Edit" link on the comment he wants to edit.

2. The system asks the user for the new content of the comment.

3. The user clicks on the "Save" button.

4. The system saves the comment.

Alternate flow: cancelling



1. The user clicks on the "Edit" link on the comment he wants to edit.

2. The system asks the user for the new content of the comment.

3. The user cancels the use case by clicking on the cancel button or anywhere but the "Save" button.

Preconditions

The document exists and it is not removed.

The comment exists. The user is the owner of the comment or is a moderator of the group or portal in which the wiki is placed.

The wiki in where they are placed has not been closed.

4.2.21. Delete a comment

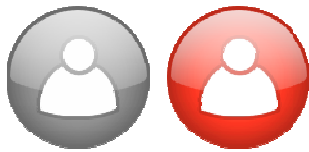
Brief description

This use case is used to delete a comment of a document.

Actors

Author of the comment or Moderator of the group or portal.

Basic flow of events



(author, moderator of the group/portal)



(System)

1. The user clicks on the "Remove" link on the comment he wants to delete.

2. The system asks for a confirmation.

3. The user confirms the operation.

4. The system deletes the comment from database.

Alternate flow: cancelling



1. The user clicks on the "Remove" link on the comment he wants to delete.

2. The system asks for a confirmation.

3. The user cancels the use case by not confirming the operation.

Preconditions

The document exists and it is not removed.

The comment exists. The user is the owner of the comment or is a moderator of the group or portal in which the wiki is placed.

The wiki in where they are placed has not been closed.



5.Design and implementation

In this chapter we will see the design for the e-Wiki and its implementation.

In the design we will see the structure of the new e-Wiki and in the implementation will see its construction.

In this chapter we will also take a look at the final result of the project, both the new engine and the running environment. We will see all of the available features of the new engine.

5.1. Technological environment

First of all let us take a look at the technological environment that involves the project.

As this engine is part of a bigger platform, it has to fit in there by trying to slow down the system as less as possible. Definitely, the most appropriate way to achieve this goal is to use and exploit the current technologies used in the platform.

In this chapter we are going to see which of these available technologies in the e-Catalunya platform have been used to develop the e-Wiki.

To ease a little the comprehension of where these technologies appear we will simulate a petition for a web from a member of e-Catalunya and we will analyze all the technologies that are involved in the way of obtaining its HTML response. The very first technology we find is the web server that listens to that request.

5.1.1. Web servers

As e-Catalunya is a web based application, it does rely on web servers to give service to its users. It uses two servers: Apache and Tomcat.

Apache

Apache is a web server notable for playing a key role in the initial growth of the World Wide Web. Apache is widely used to serve static web content. Apache has a module system which enables an easy way to integrate non-static content, such as PHP, JSP, Servlets, etc., via enabling or disabling their corresponding modules.

Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. The application is available for a wide variety of operating systems, including Unix, FreeBSD, Linux, Solaris, Novell NetWare, Mac OS X, Microsoft Windows, OS/2, TPF, and eComStation. Released under the Apache License, Apache is characterized as free software and open source software.

Since April 1996 Apache has been the most popular HTTP server on the World Wide Web.

In e-Catalunya, the Apache web server is used to serve static content. Dynamic content is better served by Tomcat, another web server.

Tomcat

Apache Tomcat is a Servlet container developed by the Apache Software Foundation. Tomcat implements the Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, and

provides a “pure Java” HTTP web server environment for Java code to run.

In e-Catalunya, the Tomcat web server is used to serve dynamic content. The main control of a request from any user of e-Catalunya is managed by Tomcat.

The following step in the simulated web request comes across the eXo platform.

5.1.2. eXo Platform: portal framework manager

eXo Platform' is an open source, open standards, enterprise scale portal, content management system. The eXo Platform is developed using Java technology.

The eXo platform is useful in e-Catalunya in the way it provides a portal based web environment so content can be separated by portals. Apart from that, eXo includes a portlet container.

Portlets are pluggable user interface components that are managed and displayed in a web portal. Portlets produce fragments of markup code that are aggregated into a portal page. Typically, a portal page is displayed as a collection of non-overlapping portlet windows, where each portlet window displays a portlet.

One of the main advantages of portlets is that they can be hot deployed.

Software deployment is all of the activities that make a software system available for use. What is known as hot deploy is the capability of a software to be deployed without requiring the restart of the entire system.

This improves the general process of developing new code by sharpening the task of testing new code.

Although portlets are hot deployable, some other parts of e-Catalunya need a restart of the web server environment in order to be reloaded. This lack is caused by some technologies used such as the data manager, services, etc.

The eXo platform passes the flow control to the main programming language used in e-Catalunya.

5.1.3. Programming languages

Several languages are used in the e-Catalunya platform. Some of them compose the resulting code and they are rendered and interpreted by the web browser. Some others are used to dynamically generate this output code. The most important one, in which is based the entire platform, is the Java programming language.

Java

Java is a programming language originally developed by Sun Microsystems and released in 1995. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. This gives us platform independence so we can manage it in different environments.

Platform independence means that programs written in the Java language run similarly on any supported hardware/operating-system platform. This allows us to have a development environment based on a Windows operating system and a production environment based on a Linux operating system.

Java is used in e-Catalunya as the main controller of the execution flow of each request from any user of e-Catalunya. We use this language to analyze users' petitions, access the database and retrieve the requested information. This information is delivered to the technology in charge of the presentation tier: Velocity by putting in context the needed variables. We will see tiers later in chapter 5.2 (Architecture).

Here is an example of how Java invokes one of those templates, called "greetUser.vm", and sets a message to be showed to the end user.

```
...
    User myUser = new User("Mike");
    context.put("user", myUser);
    Date today = new Date();
    context.put("today", today);
    return getTemplate("greetUser.vm");
}
```

Velocity

Velocity is a Java-based template engine that provides a simple yet powerful template language to reference objects defined in Java code. Its aim is to ensure clean separation between the presentation tier and business tiers in a Web application.

Apache Velocity is an open source software project directed by the Apache Software Foundation.

Velocity retrieves the information send by the business tier as Java objects and generates the output HTML code. Velocity engine also supports some basic conditional branching and looping instructions while processing the templates. Let us take a look at how the "greetUser.vm" could look like:

```
<HTML>
  <BODY>
    #if ($user.getBirthDate() == $today)
      Happy birthday, $user.getName()!
    #else
      Hello $user.getName(), how are you today?
    #end
  </BODY>
</HTML>
```

The resulting output code generated by the velocity engine is sent to the browser and displayed into it. This code is written in XHTML and uses CSS and JavaScript.

XHTML

The Extensible Hypertext Markup Language, or XHTML, is a markup language that has the same depth of expression as HTML, but also conforms to XML syntax.

HTML provides a means to describe the structure of text, based information in a document, by denoting certain text as links, headings, paragraphs, lists, and so on, and to supplement that text with interactive forms, embedded images, and other objects.

XML is a general-purpose specification for creating custom markup languages. Its primary purpose is to help information systems share structured data, particularly via the Internet.

The most important difference between XHTML and HTML is the requirement that the document must be well-formed and that all elements must be explicitly closed as required in XML. Also all element and attribute names are case-sensitive, so the XHTML approach has been to define all tag names to be lowercase.

For instance, the following HTML code:

```
<HTML>
  <HEAD>
    <TITLE>
      Welcome
    </TITLE>
  </HEAD>
  <BODY>
    Hello world!
    <BR>
  </BODY>
</HTML>
```

Would correspond to the following XHTML well formed code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html lang="ca" xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>
      Welcome
    </title>
  </head>
  <body>
    Hello world!
    <br />
  </body>
</html>
```

In order to make XHTML have a better look we use CSS.

CSS

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in a markup language. Its most common application is to style web pages written in HTML and XHTML, but the language can be applied to any kind of XML document. It is designed primarily to enable the separation of document content from document presentation. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, and reduce complexity and repetition in the structural content.

JavaScript

JavaScript is a scripting language most often used for client-side web development. JavaScript is a client side scripting language used to dynamize web-content, allowing the client to exert control on the page's contents.

JavaScript, despite the name, is essentially unrelated to the Java programming language, although both have the common C syntax, and JavaScript copies many Java names and naming conventions.

The primary use of JavaScript is to write functions that are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page. Some simple examples of the usage of this language are:

- Opening or popping up new windows.
- Validation of web form input values to make sure that they will be accepted before they are submitted to the server.
- Hiding or showing specific information at the time the user needs it.

5.1.4. Data management

Data storage in e-Catalunya is provided by MySQL, a database management system.

MySQL

MySQL is a relational database management system. It is used in e-Catalunya to store all the data introduced by members. Contents stored in this database are managed by Hibernate.

Hibernate

Hibernate is an object-relational mapping library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database.

Hibernate is free as open source software that is distributed under the GNU Lesser General Public License.

Hibernate's primary feature is mapping from Java classes to database tables (and from Java data types to SQL data types). Hibernate also provides data query and retrieval facilities.

In e-Catalunya, each Java object is mapped to database using Hibernate. This kind of translation is done by defining some XML specific files. By this, the only thing to do with database for the new e-Wiki is to define these XML files.

5.2. Architecture

e-Catalunya is implemented following some software design patterns.

As hinted before, in chapter 5.1 (Technological environment), e-Catalunya is structured in a three-tier architecture.

Three-tier architecture is a client-server architecture in which the user interface, functional process logic ("business rules"), computer data storage and data access are developed and maintained as independent modules, most often on separate platforms.

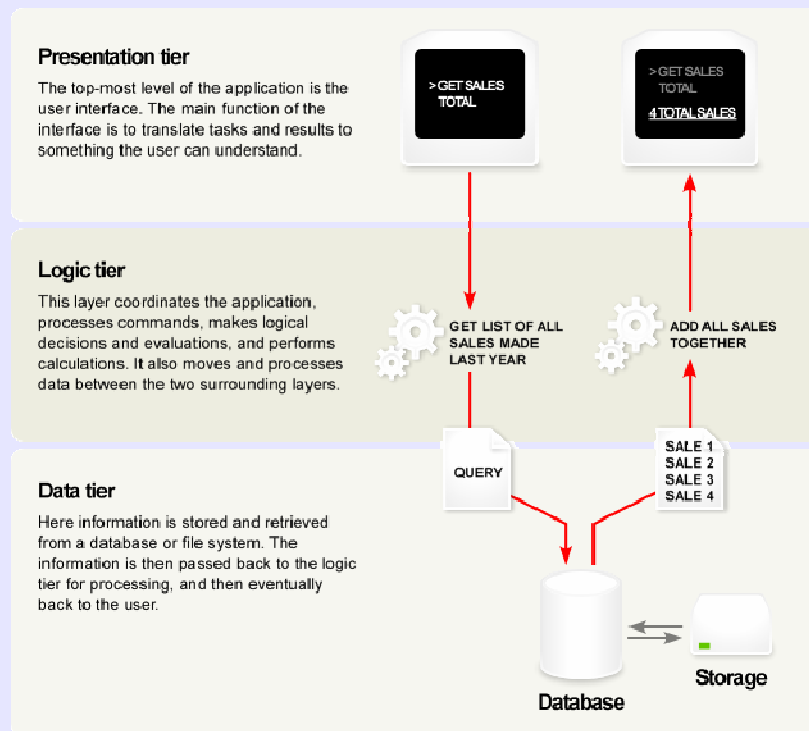


Figure 22: Visual overview of a three-tier application.

Image retrieved from Wikipedia.

Each tier is managed by different technologies.

Presentation tier

The presentation tier is the one that interacts with the final user. Is responsible of to attend user's petitions, retrieve the needed

information and show them the requested information. The presentation tier relies on eXo platform as a portlet container. In this layer we find another design pattern: model-view-controller.

Model-view-controller pattern isolates business logic from user interface considerations, resulting in an application where it is easier to modify either the visual appearance of the application or the underlying business rules without affecting the other.

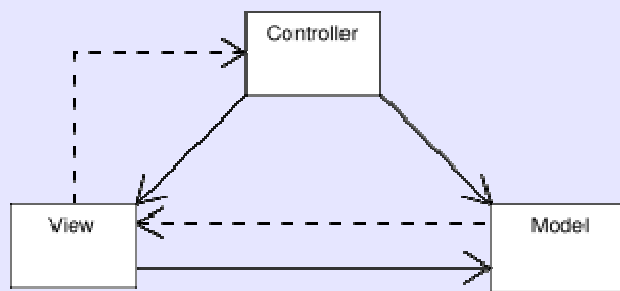


Figure 23: Visual overview of a model-view-controller pattern.

Image retrieved from Wikipedia

The control is managed by portlets, the view is managed by the velocity template engine and the model is managed by the logic tier of the system architecture.

Logic tier

The logic tier is pulled out from the presentation tier and, as its own layer, it controls an application's functionality by performing detailed processing. We will see the model of the e-Wiki later, on chapter 5.3 (Conceptual model).

Data tier

Data tier consists of Database Servers. Here information is stored and retrieved. This tier keeps data neutral and independent from application servers or business logic. Giving data its own tier also improves scalability and performance. Data tier of e-Catalunya is

managed by Hibernate, which stores all the data into the MySQL database.

5.3. Conceptual model

In this section we are going to see the class diagram for the e-Wiki.

A class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes.

e-Wiki's class diagram is simplest enough to be shown entirely in a single figure:

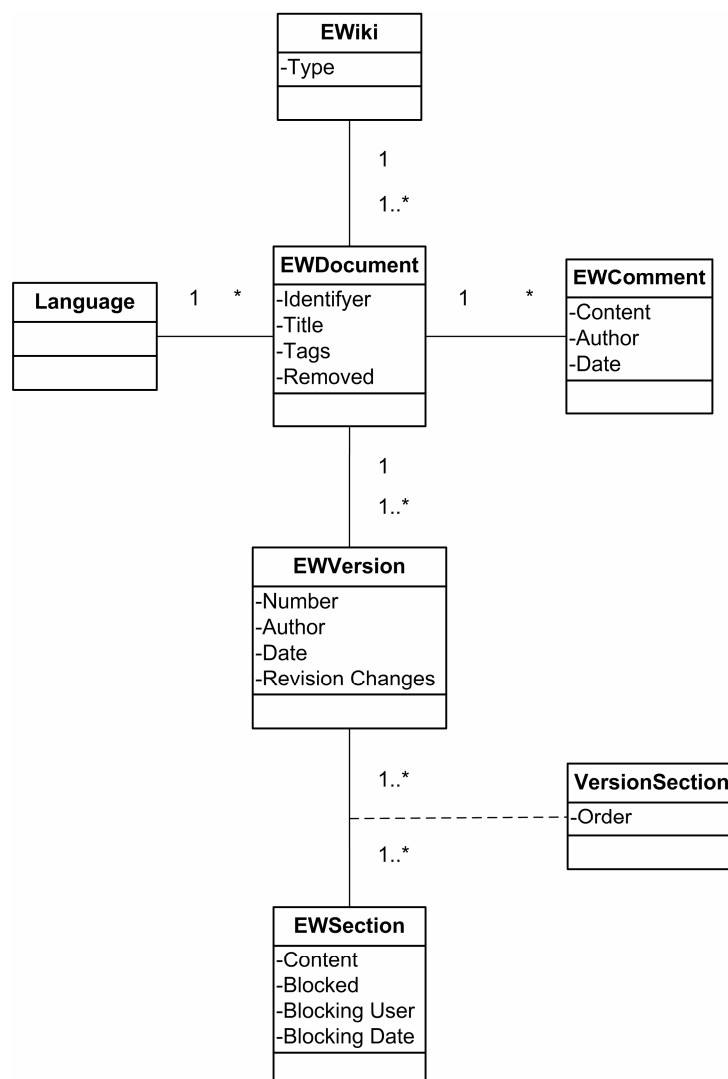
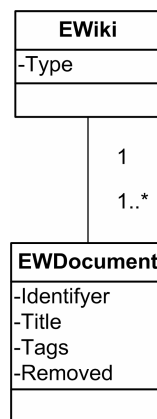


Figure 24: class diagram of the e-Wiki.

In this figure we can see a general overview at which entities appear in the wiki engine. First of all, in the top of the structure,

there's the "EWiki" entity. Each instance of this entity represents a single wiki tool placed in a group or a portal. It has one attribute called "type". This attribute does not mean anything at all in the e-Wiki scope, as it only tells the type of the wiki. This attribute is used later, in the e-Catalunya scope, where tools can be of many types. This attribute will be mapped to those types.

Connected to the wiki, there is the EWDocument entity.



This entity represents a document. Their relationship is one-to-many, as one wiki can be related to many documents but each document can only be related to one wiki.

Relationships between entities have one cardinality number for each entity involved. These numbers indicate the minimum and the maximum instances of each entity that can appear in the relationship. This cardinality numbers use a specific nomenclature:

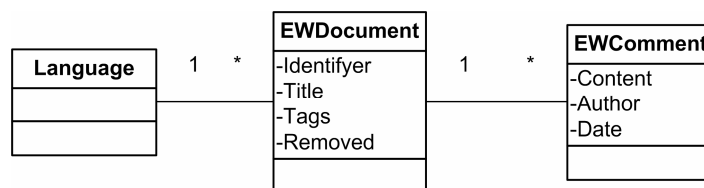
"0..1": means that it can appear 0 or 1 instance of the entity at maximum.

"1": means that it must appear 1 instance of the entity by force.

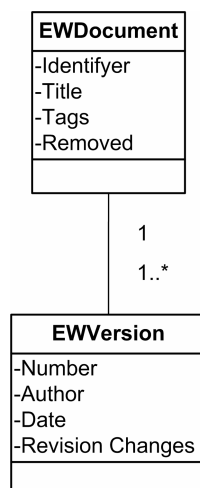
"0.." (or "0..n" or simply "*" or "n"): means that it can appear any number of instances of that entity, this is from 0 to infinity.*

"1.." (or "1..n"): means that it must appear at least one instance of that entity.*

Each document has an identifier, a title, some tag words and a removed flag. Each document can have as many comments as desired and is related to a language meaning that it is written in that language. The language entity still exists in the platform. Due to this, the relationship between them has been implemented only as an attribute: a locale, representing the language.



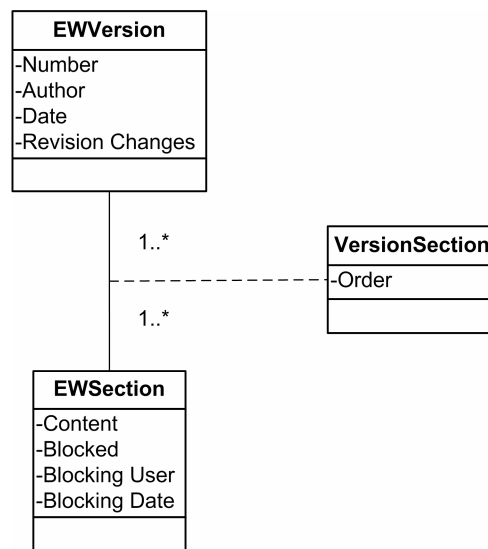
Following the requirements, users want to maintain a history of the course of a document. To maintain this history we have the EWVersion entity, which represents a version of a document. The relationship between documents and versions has been implemented as one-to-many, as one document can have many versions, but a version belongs to a single document.



Each version has a number. The first version of a document will have number 1; the second, number 2, and so on. When someone

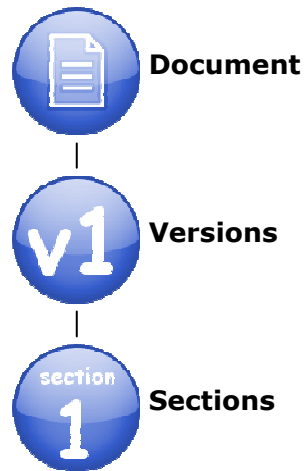
edits a document and saves it, he is implicitly creating a new version of that document. That version has an author, and a date of creation, which in fact represents the date of the last modification of the document. The author is able to make a brief explanation about what are the differences between the version he is saving and the latest one. This information is known as “revision changes”. All this information is stored in this entity.


But then, where is the content? One of the initial requirements was to enable section editing. Section editing would let users to edit just a little section, in order to have different users editing the same document, but different sections, simultaneously. Then the content is stored in the EWSection entity, which represents each section of a document.



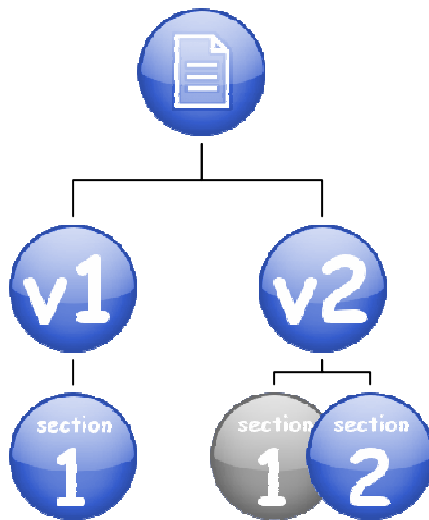
Then we have that an instance of a document has multiple instances of versions, and each instance of these versions can be related to multiple instances of sections. We could have implemented this one also as another many-to-one relationship. This would work, but wouldn't be correct enough, as it would waste much space in database, because we would be replicating much information. Let us


imagine that some one creates a document with only one section.
This would generate the following instances into the database:



 : Each ball represents an instance into database

Now let us imagine this person edits the document and adds another section to it. This would create another version of the document including the new section 2, and replicating the old one, section 1:



 : Each grey ball represents a replicate instance into database

If we repeat the operation many times, we can see that the amount of replicated sections grows exponentially...

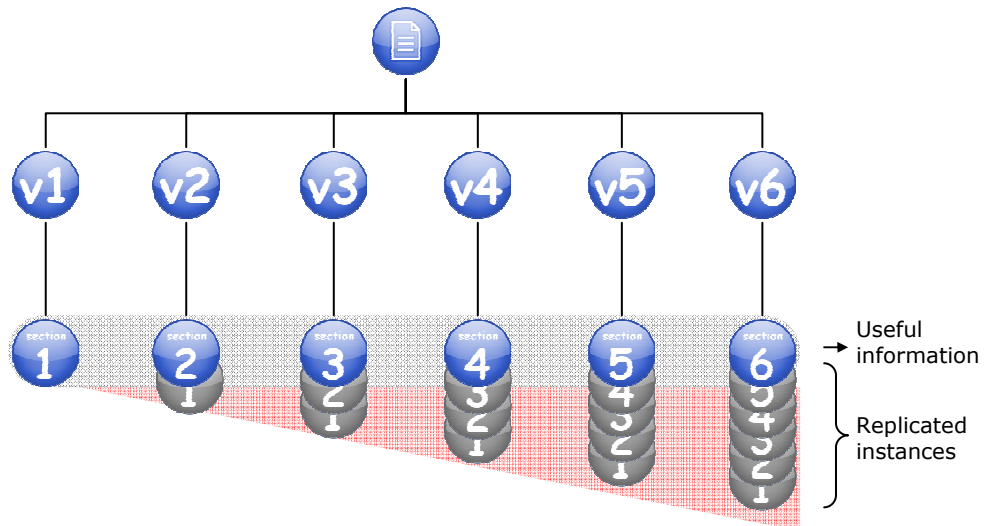


Figure 25: this would be the result of a many-to-one relationship between sections and versions.

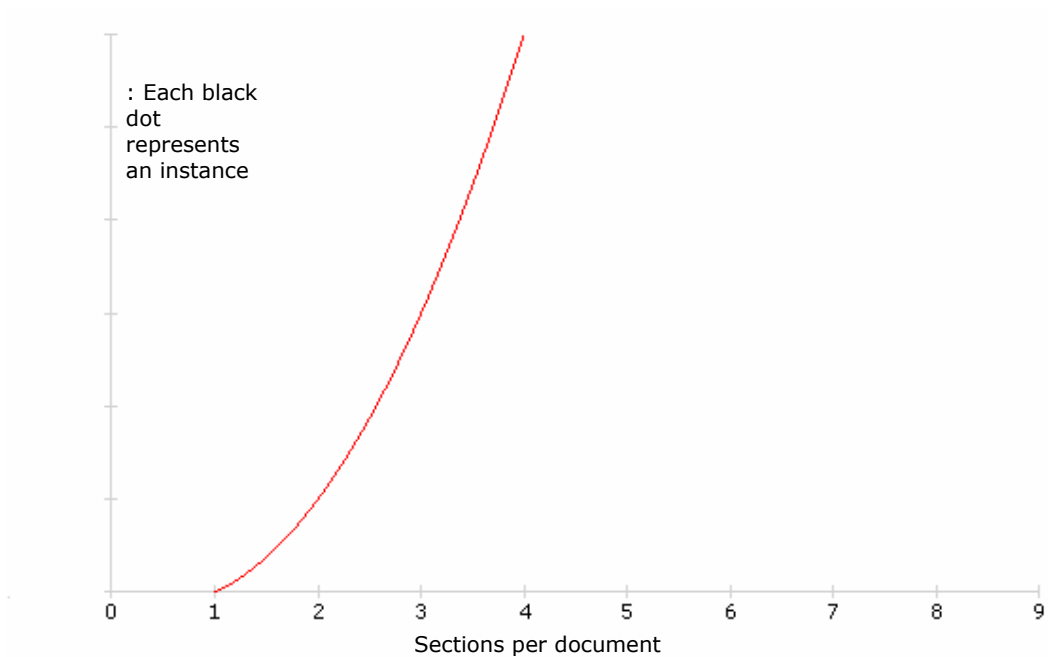


Figure 26: Graphic showing the growth of duplicate instances of sections per each new section added to a document

That's why we have implemented it as a many-to-many relationship. In a many-to-many relationship it does appear what is known as an associated class which contains extra information. In our case, this associated class is relating both versions and sections together. The extra information appearing in this class is the order of appearance of sections in each version:

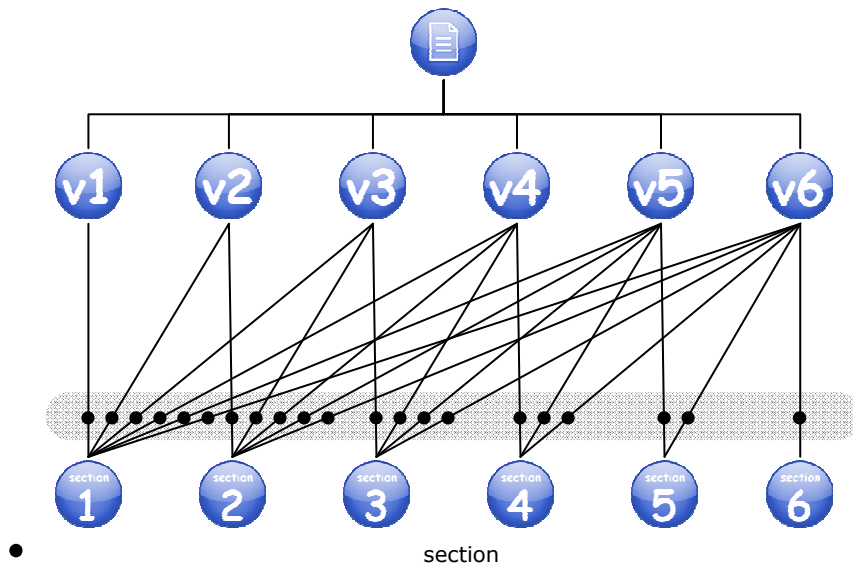
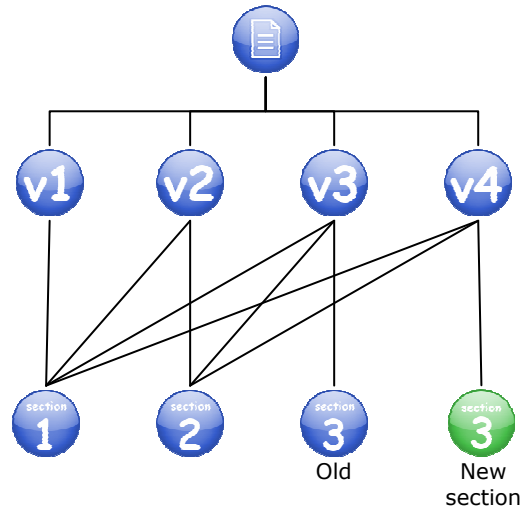


Figure 27: this is a schema of a many-to-many implementation of the relationship between sections and versions

By this way, we have the same amount of instances into database, or even some more, but they are much less heavy, because the content of the documents is stored in the sections entity, which are not replicated in database. This mechanism ensures that every single section only appears once in the database.

This can induce to think about what would happen if someone edits an existing section. As there are no duplicates in the database, what will happen to the old section? The system is supposed to maintain the history of a document so it should maintain both, the old section and the newer one. In the process of editing an existing section of the document it gets substituted by the newer one, maintaining the old one and its relationship between its own version. For instance, lets suppose that someone wants to edit the section 3 of the document showed in the example. As soon as this person saves the updated section, a newer version of the document is created with the changes done to the section. The database would be in this state:

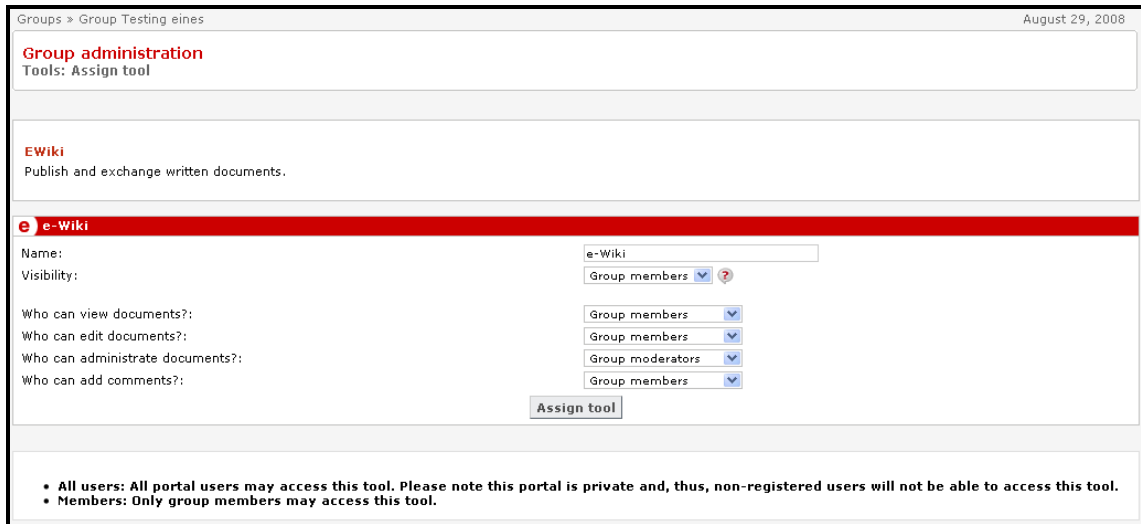


The process of editing an existing section should be viewed as a creation of a new section which will substitute an older one.

5.4. Final result

Here we are going to see the final result of the FMP. We will navigate across all of the available features in the new e-Wiki engine.

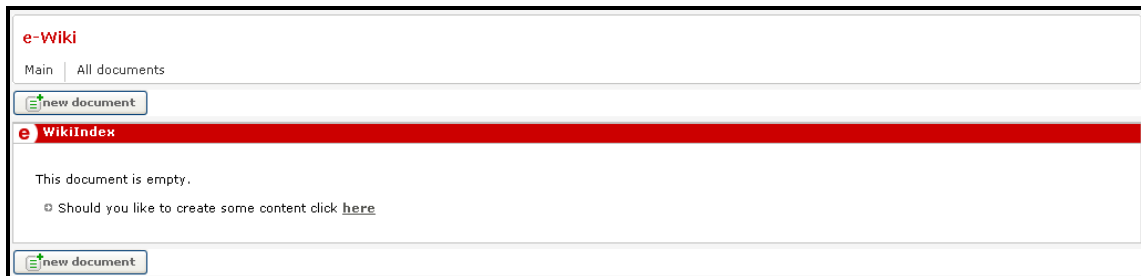
First of all, let us start by creating a new wiki tool in a group.



The screenshot shows a web interface for assigning a tool to a group. At the top, it says 'Groups > Group Testing eines' and 'August 29, 2008'. Below that, there's a section for 'Group administration' with a 'Tools: Assign tool' button. The main content area is titled 'EWiki' with the description 'Publish and exchange written documents.' Below this is a red header for 'e-Wiki'. The form contains several fields: 'Name:' with a text input containing 'e-Wiki'; 'Visibility:' with a dropdown menu set to 'Group members'; 'Who can view documents?:' with a dropdown menu set to 'Group members'; 'Who can edit documents?:' with a dropdown menu set to 'Group members'; 'Who can administrate documents?:' with a dropdown menu set to 'Group moderators'; and 'Who can add comments?:' with a dropdown menu set to 'Group members'. There is an 'Assign tool' button at the bottom of the form. Below the form, there are two bullet points: '• All users: All portal users may access this tool. Please note this portal is private and, thus, non-registered users will not be able to access this tool.' and '• Members: Only group members may access this tool.'

Figure 28: assigning an e-Wiki to a group.

After that we are ready to start working with it. The very first thing we will see when accessing it is this message:



The screenshot shows the e-Wiki interface. At the top, it says 'e-Wiki' and 'Main | All documents'. Below that, there's a 'new document' button. The main content area is titled 'WikiIndex' and contains the message 'This document is empty.' followed by a link: 'Should you like to create some content click [here](#)'. At the bottom, there's another 'new document' button.

Figure 29: accessing a non existing document

By clicking on the provided link we go directly to the edit page:

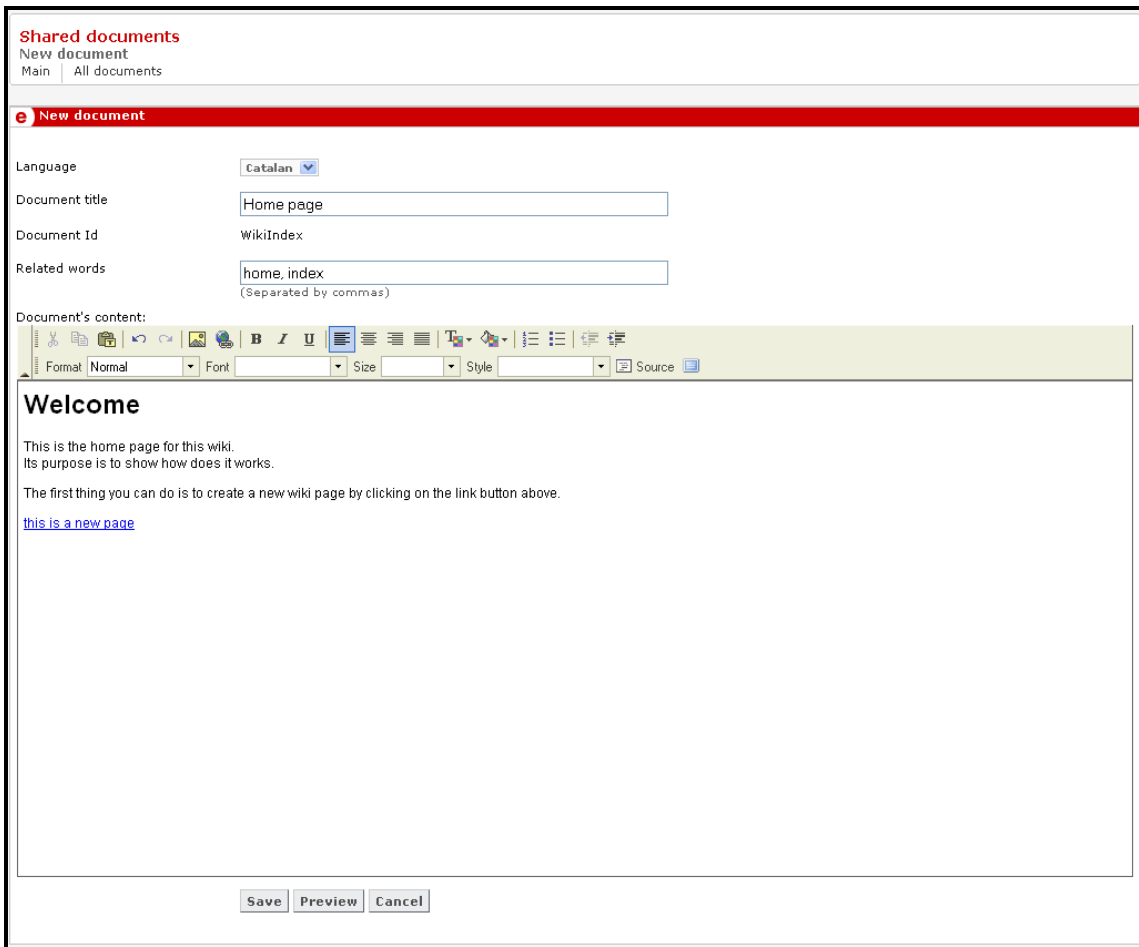


Figure 30: creating a new document

As shown in Figure 30, we can select the language of the document, the title and its related words, but not its identifier. That's because we clicked on the link in order to create a specific document, in the case: the home document. The home document is noted for having the identifier set to "WikiIndex".

The edit box is made of an open source tool called "FCKEditor", which has been tuned to fit our needs. The changes done to this editor rely basically on the "create a link" button. Wikis' principal characteristic is the easy way they have to create new pages. To achieve this, we modified the generic link button, adding the possibility to add a "wiki link".

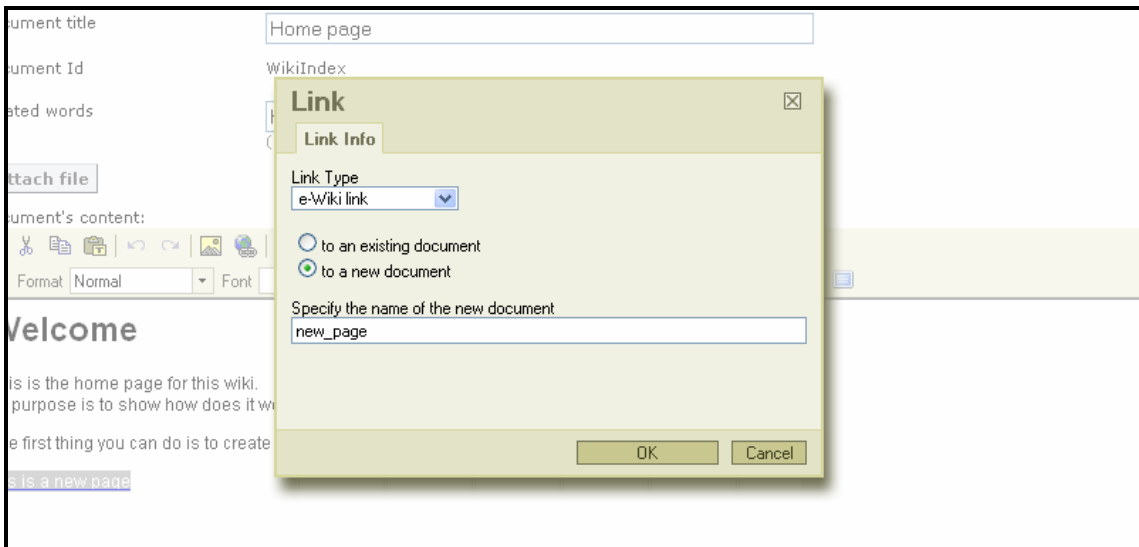


Figure 31: creating a wiki link

As shown in Figure 31, we can create a new page by selecting the appropriate option. We can choose either linking to an existing document or to a new one. If we chose to link to an existing document we are presented a list with the available documents. If there isn't any document yet in the wiki, then the list will be empty. If we chose the option "link to a new document" then we have to specify the desired name of the document.

In this step the document we are creating still does not exist. It will get materialized when clicking on the "Save" button. Before that, we can make a preview, to see how the document will look like:

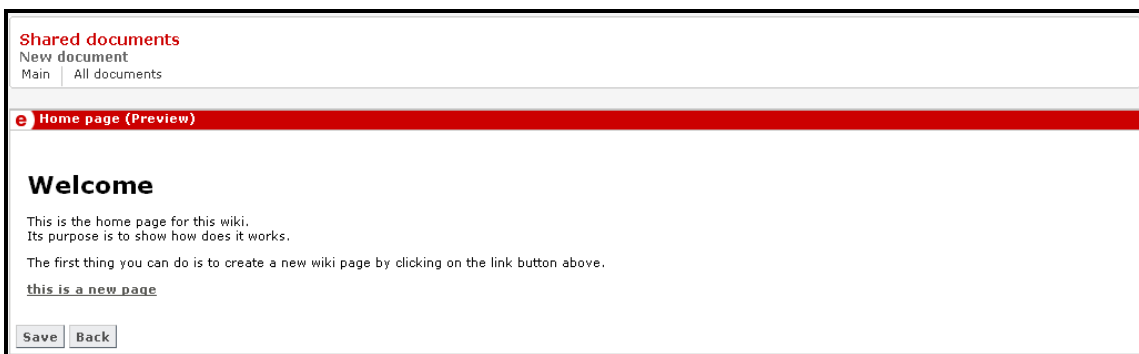


Figure 32: previewing a document

Then, if it looks fine for us, we can save the document:

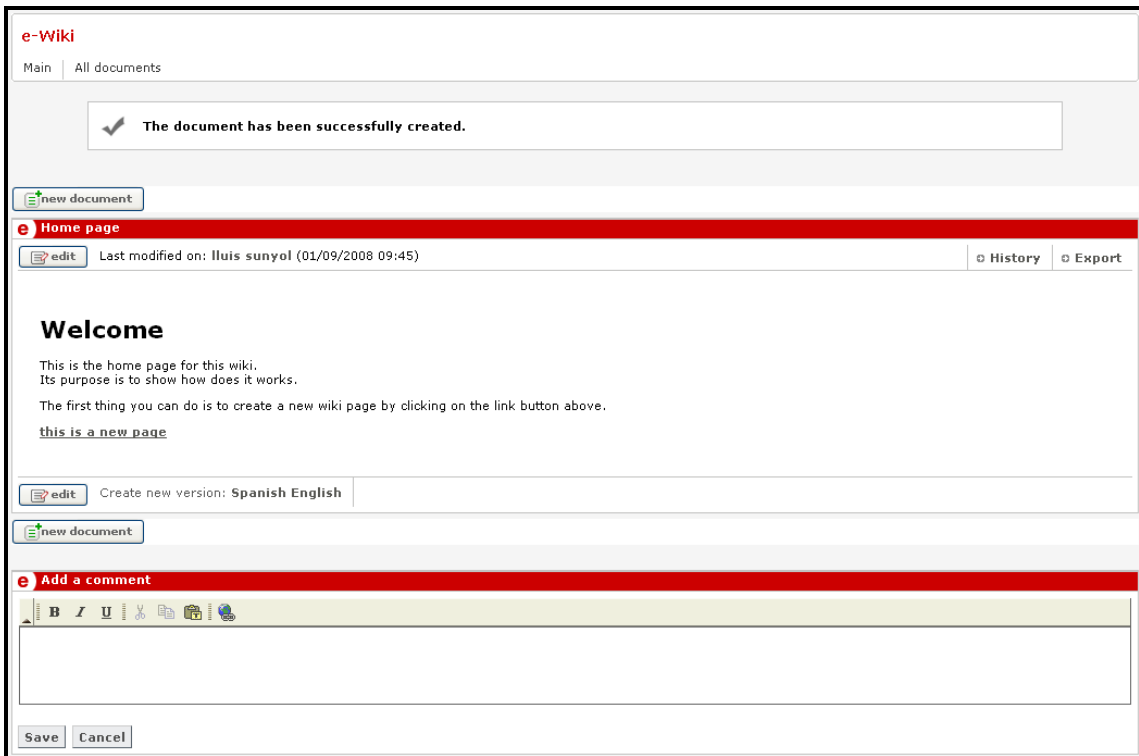


Figure 33: saving a document

Once done this, we can access the document through the documents list page, by clicking on the "All documents" link, in the tool's top menu:



Figure 34: listing all the existing documents

In the list we can find useful information. Here we will be able to see if a document is being edited, or if it has been modified recently. We can also filter the list, by only showing documents written in a specific language. As we can see in Figure 34, there is only one document in the recently created wiki: the home document. In that document we created a link to a new document, but the document which is pointing at still has not been created. This means

that it still does not exist, and so, it does not appear in the list of existing documents.

If we want to read the home page document we just have to click on it:

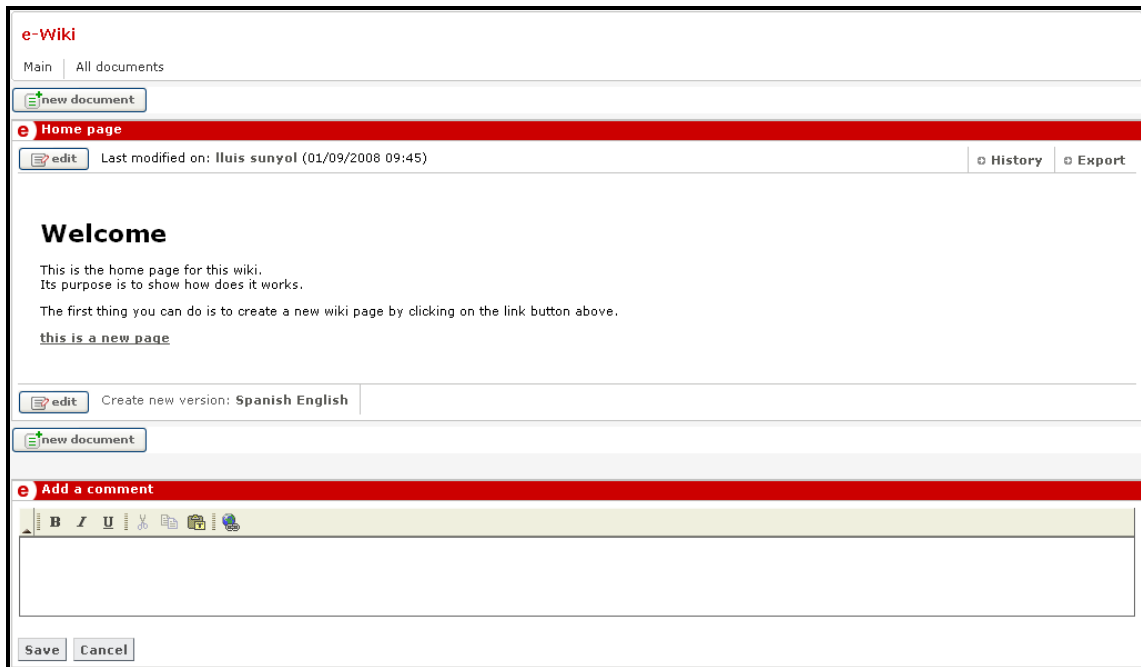


Figure 35: viewing a document

In this point we can make comments, edit the page, export it, or view its history. If we edit the page, we get the same page as when we were creating it, with some differences: when editing a document we are suggested to enter the purpose of the edition, and also we are permitted to attach files:

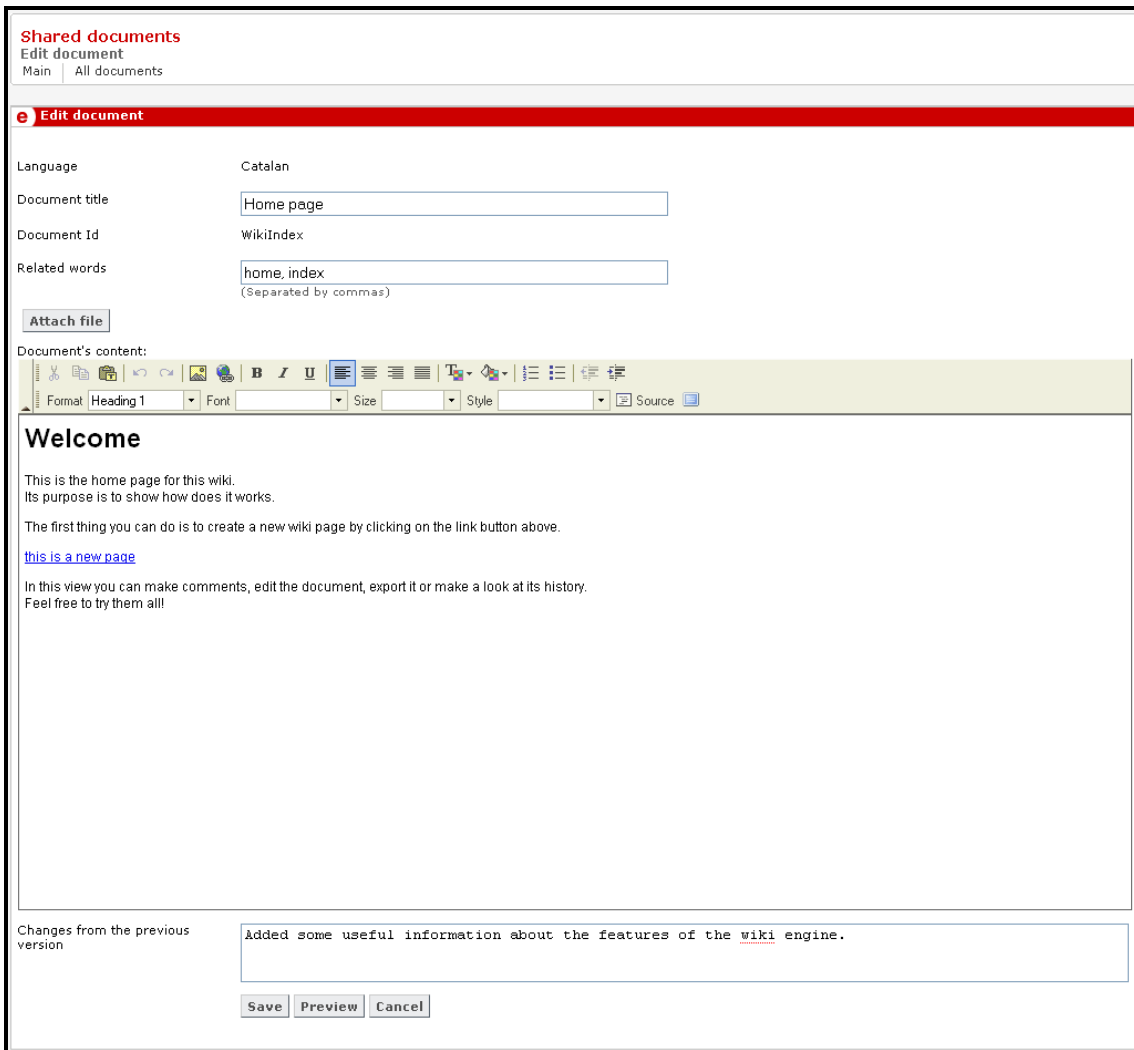


Figure 36: editing an existing document

When editing a page, we can change almost all of its contents. The only ones that can not be changed are the identifier and the language. They can not be changed⁵ here because they both identify a single document inside a wiki.

⁵ In fact this is not really true. The identifier can be changed in the listing documents page. It will be seen later.

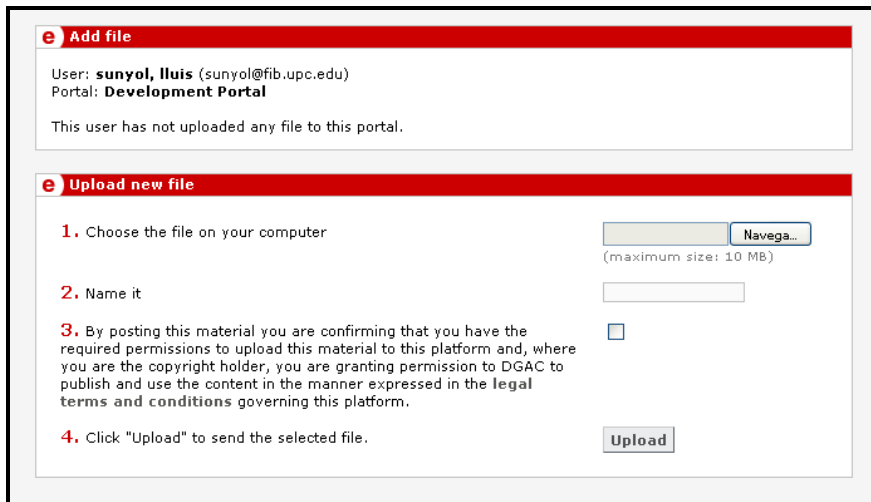


Figure 37: attaching files to a document

Let us add some phrases and save it again.

Now, if we go back and check the history of the document we will find that the document has 2 versions, because we edited it twice.

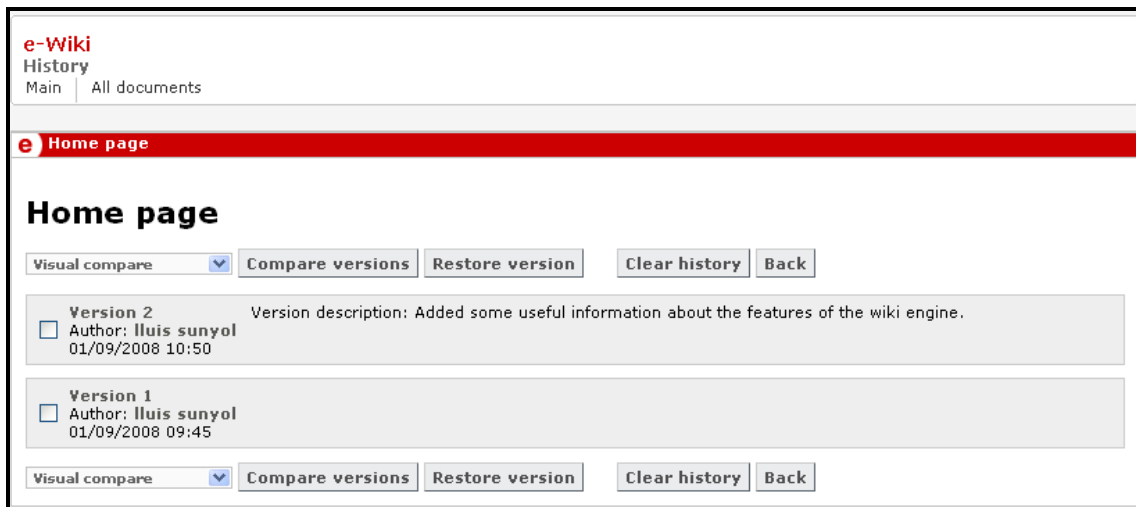


Figure 38: viewing the history of a document

In this page we can compare two versions of the document, restore one of those versions or clear the history of the document.

We dispose of two different kinds of comparisons: visual comparison and source code comparison. Visual comparison is designed to usual members. It shows both versions side by side:

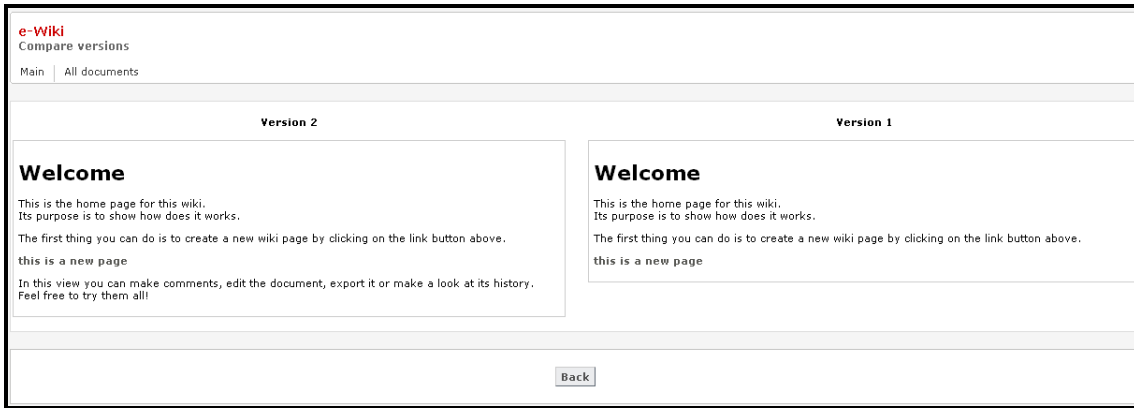


Figure 39: visual compare of two versions of a document

Source code comparison is designed for those members with advanced knowledge of web technologies, specifically HTML. It shows the differences between the HTML codes of both versions.



Figure 40: source code comparison of two versions of a document

Let us get back to the listing documents page. There appeared three disabled buttons. They were disabled because no document was selected. As soon as we select a document they get enabled:



We can remove it, change its identifier or unlock it.

Changing the identifier of a document will affect currently related link to the document, so it is desirable to use this feature as less as possible.

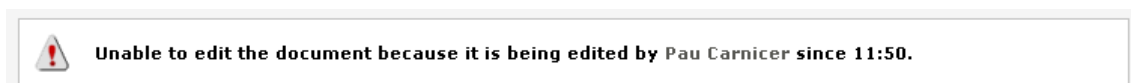
Unlocking a document is useful if someone who clicked on the edit button of a document has left the process without cancelling it, but this is a double-edged sword, because it can cause some concurrency problems. In any case, the system never loses any information. If there's any concurrency problem, the affected user is warned about that and it's provided with a solution.

Removing a document puts it on a removed state in which it can't be edited. It can only be read, restored or deleted by a member with the corresponding permissions.



Figure 41: removed document

Let us suppose we want to edit again the home document but, unfortunately, a friend of us is editing it at the moment. The system will warn us with a message like this one:



If we check the listing documents page we will notice the lock on the home document, meaning it is locked to edit it:

Shared documents					
<input type="checkbox"/>	<u>Document title</u>	<u>Language Id</u>		<u>Versions</u>	<u>Last modified on</u>
<input type="checkbox"/>	Home page	Catalan	WikiIndex	2	lluis sunyol (01/09/2008 10:50)
<input type="button" value="Remove"/> <input type="button" value="Change Id"/> <input type="button" value="Unlock"/>					
<input type="checkbox"/> documents being edited <input type="checkbox"/> documents modified recently					

Figure 42: locked document

If we wanted to edit the document we should wait until it gets saved or cancelled. If we think that the user who appears editing the document has left, we can wait until the lease time expires, which is fixed to 30 minutes, or we can accelerate this process by unlocking the document manually, so it becomes available again.

These are the most important actions members can interact with the new wiki engine.

5.5. Development environment

In this section we are going to see the environment in which this project has been developed. All of the software used to develop the project is listed below.

Each developer in the e-Catalunya team is provided a complete development environment. This environment consists of a workstation with the following technologies installed:

Windows XP

Windows XP is the operating system used in the whole LCFIB. As e-Catalunya is platform independent, there is no problem on developing it under a windows operating system while running it in a Linux operating system, in other environments.

Tomcat

Differing from any other environments of the platform, in our local machines we don't have installed the Apache web server; we only dispose of a Tomcat web server. This makes tomcat to serve not only dynamic but also static content. This could make the system a little slower, but as it is a local environment, it is not expected to carry much load from users, as only one developer is working with it, probably most of times will only have to serve one petition at a time.

MySQL

A MySQL server is installed in our integration environment. Our local databases are located there. Although it is not installed in our load workstation, we dispose of a complete database for our testing purposes. That's why we always need to have a connection established with the integration environment.

Eclipse

This is the software used in the whole platform to develop Java code. Although is referred many times as a Java editor, it is more than that. It provides a complete environment for developing, testing and even debugging java web applications.

Microsoft Visual SourceSafe

Microsoft Visual SourceSafe (VSS) is a source control software package oriented towards small software development projects. Like most source control systems, SourceSafe creates a virtual library of computer files. Users can read any of the files in the library at any time, but in order to change them, they must first "check out" the file. They are then allowed to modify the file and finally check it back in. The changes are made available to the other users only after the file has been checked in. Thus, a file cannot be edited by multiple users simultaneously.

This system has a plug-in for eclipse which enables an easy way to check out and edit files.

Web developer

This technology is a plug-in for the Firefox web browser. This plug-in enables an easy way to test and debug the presentation tier, acting directly with the browser.

5.6. *Running environments*

e-Catalunya was deployed for the first time in 2005 and is currently running its 1.7 version.

Of course, new releases are not developed in the same environment as final members interact with. e-Catalunya has up to four different environments in order to enable developers to develop and test new features without involving final users. These environments are, from least to most important: development, integration, pre-production and production.

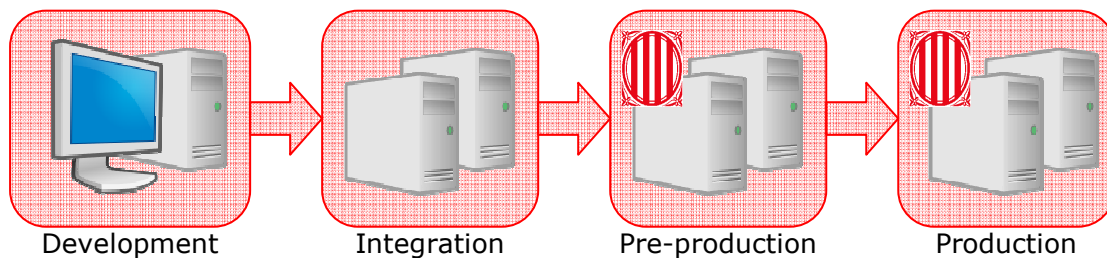


Figure 43: e-Catalunya lifecycle of new features

- **Development:** this is the first step in the way to being released to the final users. It is also the simplest environment in e-Catalunya. It consist of a single personal computer connected to a database of its own, hosted in the integration environment. Each developer has his own *development environment*. This let us develop and test new features without involving other developers work. Concurrency control in this environment is done by the Visual Source Safe software (VSS). When a new feature is developed, it steps forward to the integration environment.

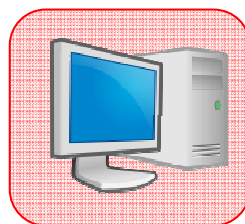


Figure 44: development environment

- **Integration:** this environment consists of two machines placed inside the LCFIB. The main objective of this environment is to test new features and find possible errors. Errors found in this phase have to be solved and updated again to this environment before stepping forward. When no more errors are found in a new feature, it is included on the next release of the e-Catalunya platform. Periodically, these new releases are loaded to pre-production environment.

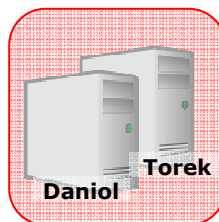


Figure 45: integration environment

These two servers in the integration environment have their own names. They are called "Torek" and "Daniol".

- **Pre-production:** this environment is a replica of the production environment. It's placed at the company contracted by Generalitat de Catalunya to provide hosting infrastructure. Its objective is to emulate as best as possible the behaviour of new features on the production environment in order to detect possible errors. It's the last step of the way of being offered to the final users. New features arriving to this environment are expected to not to fail in any way. If any problem is detected in a new feature of a release in this environment it usually causes for the new feature do be discarded. Anyway, the problems must be solved (by discarding the feature or by solving it) before stepping forward to the production environment.



Figure 46: pre-production environment

- **Production:** this environment is the one used by the final users of e-Catalunya. It consists of two balanced server servers placed at the company contracted to provide hosting infrastructure. No errors should be found in this environment. If there appear errors in this environment, users will suffer them before a new release of e-Catalunya is loaded.



Figure 47: production environment

The expected lifecycle of a new feature is to start being developed in a development environment and step forward to the production environment.

5.7. Development, testing and deployment

Before the development of new code ended, some tests were done to the e-Wiki in order to improve the way of interacting with it. Some **usability tests** were done and their results were applied for the first release of the engine.

The development of new code was completed in **early July**. Then it was deployed in our integrating environment and was started a fully **testing phase**, by part of the e-Catalunya team. This phase consist of testing all of the possible inputs the system offers. Many errors are usually found in this phase. These errors are classified from the most critical to the less ones, and are solved in this order.

When the engine arrived to a stable version, it was **deployed on the pre-production** environment, checking the whole system works, to ensure the final deployment of the e-Catalunya platform won't fail when deploying it to the production environment.

Finally, with the pre-production environment working with the last version of e-Catalunya without major problems, it was **deployed on the production environment**. This happened in the **half of July**.

Nowadays, e-Catalunya is running the first version of the e-Wiki. The old wiki engine, based on the XWiki software, can not be used any more to create new instances of wikis, but the engine is still serving old pages. A migration task has started in order to be able to completely remove this old engine as sooner as possible. The new e-Wiki engine is available to e-Catalunya members.



6. Project conclusions

This chapter concludes this final master project.
A cost analysis follows.

6.1. Cost Analysis

The cost of the whole project is divided in 3 separate parts: human resources cost, hardware costs and 3rd party software costs.

Human resources cost

The following table details the cost of the human resources used in this project.

Phase	Worker type	Price/hour	Hours	Cost
Study of wiki engines	System analyst	35 €	390	13.650 €
Analysis and design	System analyst	35 €	220	7.700 €
Development	Developer	25 €	380	9.500 €
Total			990	30.850 €

Hardware costs

Hardware costs include 2 servers for system operation and an additional computer for development tasks:

Hardware	Quantity	Price	Cost
Personal computer	1	1.000 €	1.000 €
Analysis and design	2	4.500 €	9.000 €
Total			10.000€

Software costs

All the software used in this project is free software, and in some cases it is also open source software, so it has no additional cost.

Total costs

This is the total cost of this project:

Resource	Cost
Human resources	30.850 €
Hardware	10.000 €
Software	0 €
Total	40.850€

6.2. Future improvement proposals

This FMP has consisted in developing a new wiki engine in the e-Catalunya platform. Future improvements can be divided into two categories: those related directly with the features of the wiki engine and those related to the behaviour and interaction with the rest of the platform in which has been developed.

Wiki improvements

In the beginning of the analysis of the new e-Wiki we studied all of the possible features we could include in the engine. From the approved ones, most of them where also expected to be released in the first version of the e-Wiki. The others are expecting to be developed in the future.

One of them is the implementation of sections inside documents. The design of the whole project has taken care of them, but the implementation has not. This means that the system is ready to put up with sections, but final users have no way to use section editing.

The exporting tool is also expected to expand to other formats, including in first instance PDF (portable document format), and in a second way ODF (open document format).

Another minor improvement expecting to be developed is to make the engine a little bit more customizable, letting the users to customize the name of the home page and the lease time given when editing a document.

Platform improvements

In the current version of the e-Wiki we have implemented the possibility to attach files to documents. By internal configuration, unfortunately files can only be attached once the document has been saved for the first time, never before. To let users attach files even

when the document still hasn't been ever saved some internal changes have to be done to the engine.

One of the future improvements of the whole platform is to enhance interaction between different tools. This involves not only the e-Wiki, but also the other tools e-Catalunya has.

Finally, another important future improvement is to remove the XWiki engine definitely from the e-Catalunya platform. As the wiki was the last remaining XWiki based tool, now it can be completely removed. But before that, all of the existing old wikis have to be migrated to the newer one. This migration task has already begun.

6.3. Final master project conclusions

The main reason of this FMP was, more than updating an outdated tool; creating a new powerful and easy to use wiki tool, expected to make e-Catalunya members wish to use it, create their own documents and share knowledge with the community.

This main goal has been achieved. A new wiki engine is now available to the members of the platform since the 16th of July of 2008, when it was released the version 1.7 of the e-Catalunya platform.

The new engine compiles with security issues, and completely surpasses his predecessor in reliability, usability, accessibility and integration.

The engine is faster and consumes fewer resources, as it exploits the available technologies used in the platform. It has a better usability, as it has passed many tests, and it offers much more possibilities than the old engine. It is fully integrated in the platform as a new tool available for being assigned.

In the overall, enhances the collaborative framework provided by the whole e-Catalunya platform.

6.4. Personal conclusions

Developing this project has given me the experience on how does the course of a project evolve, from its beginning, in analysis and design phases, until its deployment.

Many of my skills have been improved in the experience scope. In many of them I only had theoretical knowledge. This project has added the experience of how they usually work, which problems are usually found and how they are usually solved.

Team working in e-Catalunya platform has been a great experience.

By working in the e-Catalunya project in general, one notices that the communication in a project is one of the most valuable factors in its evolution. Identifying users needs, suggesting ideas, hear other's opinions, evaluate them, discuss them and decide which one is the best option, the best solution, are, in my opinion, some of the most valuable skills a developer can have. Helping and getting helped when needed it is crucial to ensure the future of a project.

e-Catalunya is a large platform, and so is the code that supports it. Learning the internal behaviour of a whole project which is running from years is not easy, and it requires special dedication in investigation and knowledge of the technologies and methodologies used in the platform.

It has also been very interesting to work with people outside the e-Catalunya development team. Having feedback from e-Catalunya users, from usability experts, from testers, etc. makes you notice how to improve your work, and gives you the possibility to develop a professional project.

Finally working inside the LCFIB has been a great experience. Internal presentations have been done during the course of this final master project which gave me the experience in how to show my

project in an enjoyable way, capturing audience's attention for a short period enough to make them curious about it.

6.5. Acknowledges

Last but not least, I would like to thank a lot of people for their help, support and collaboration during the course of this project and during my stay here in the LCFIB.

I would firstly thank all of my workmates: Laia Avilés, Pau Carnicer, Alba Coll, Daniel Golobart, Rosa M^a Martín, Albert Miró, Lucas Ponce and Hector Puente for all the help, in both professional and personal scopes.

Thank the rest of the LCFIB staff for providing those break times having fun, laughing at almost every possible thing going on in the LCFIB.

Finally I would also like to thank all of my friends and my family for supporting me and helping me in so many different ways, to achieve not only this project goal, but also all of my stay in this university.



7. Bibliography

Wiki engines study and comparison:

<http://www.wikimatrix.org>

Wiki engines (specifications and sandboxes):

<http://www.dokuwiki.org> (Doku Wiki)

<http://www.mediawiki.org> (MediaWiki)

<http://twiki.org> (TWiki)

<http://moinmoin.wikiwikiweb.de> (Moin moin)

<http://www.pmwiki.org> (PMWiki)

<http://phpwiki.sourceforge.net> (PHPWiki)

<http://info.tikiwiki.org> (TikiWiki)

http://cocondev.org/daisydocs-1_3/daisywiki (Daisy Wiki)

<http://www.xwiki.org> (XWiki)

Java specifications

<http://java.sun.com>

Hibernate

<http://www.hibernate.org>

Velocity:

<http://velocity.apache.org>

HTML, CSS, Javascript and most “presentation” related features:

<http://www.w3schools.com>

FCKEditor:

<http://www.fckeditor.net>

Web Accessibility initiative:

<http://www.w3c.es>

Open source initiative:

<http://www.opensource.org>

Wikipedia:

<http://en.wikipedia.org>