- BABÉ Louis Rémi
- GARNIER Samuel
- GONNET Clément

Table of content

1. Introduction - Users and uses

The aim of our project is to create an online presentation program bound to academics and students of the University of Teesside. A presentation (also referred as *slideshow*) is a set of slides (also referred as *transparency*) which contains elements such as text, images and charts. Slides are presented one after the other to an audience for an instructional purpose.

Academics will be able to log in the application, create presentations, manage or edit existing ones, and make them available for students. Students will be able to log in the application, browse through the presentations available for them (according to the course they take part in), preview them within the application and download them as an HTML file.

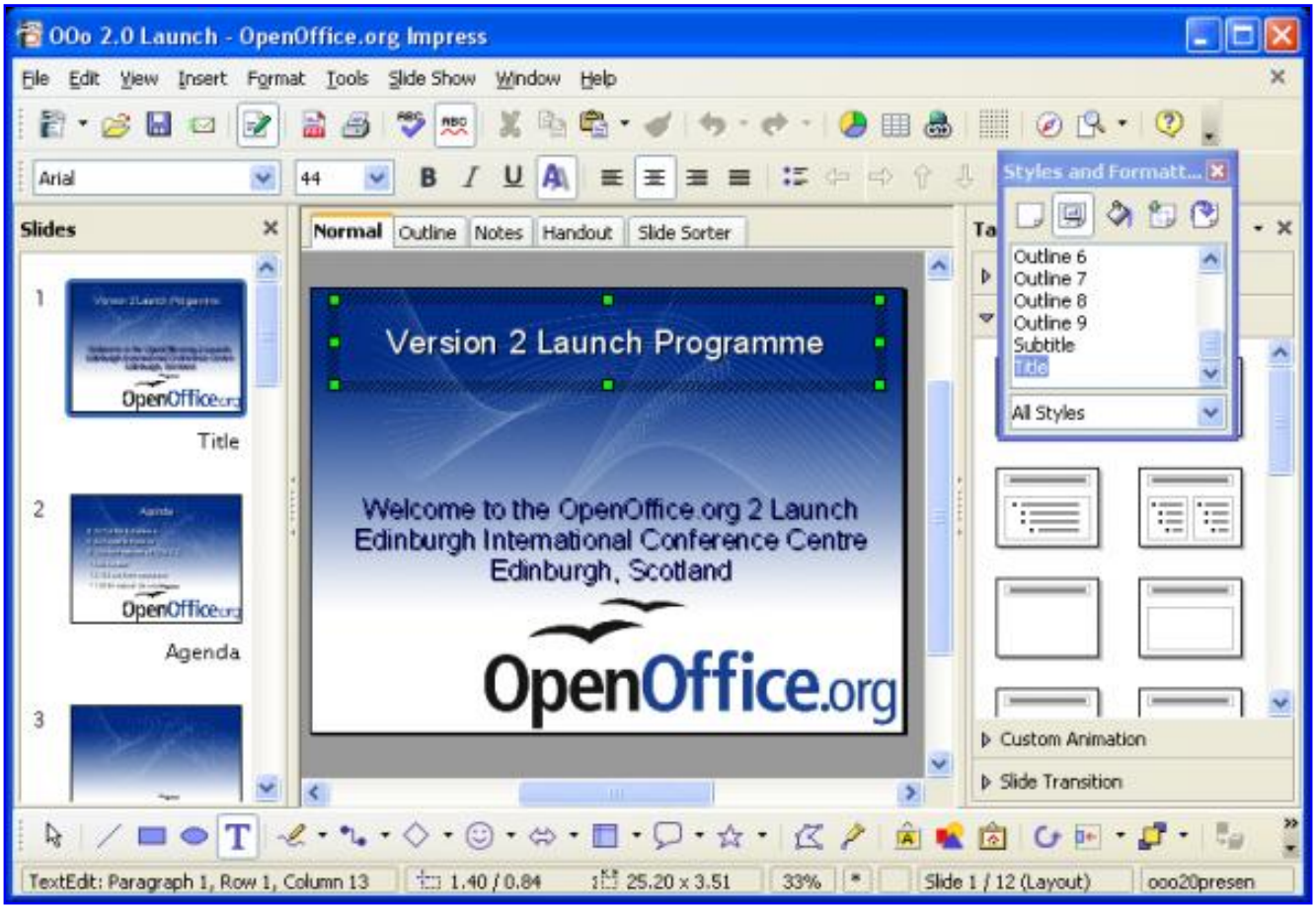
2. Why another presentation program ?

There is already, among desktop as well as web application, many existing solutions to create presentations

2.1. Desktop full-featured applications

The most known of those softwares are the desktop ones, such as Microsoft PowerPoint and OpenOffice.org Impress. Their particularity is to have a large amount of features (and, coming hands in hands, a large amount of toolbars, menus and sub-menus), allowing to customize each and every part of the slides : the transitions between slides, the animations, size, color, shape, inclination, ... of the smallest element of the slide.
They are appropriated to produce professional presentation, corresponding exactly to user's needs and desires. Until now they are the best tools for *physical* presentations, as they allows to have a

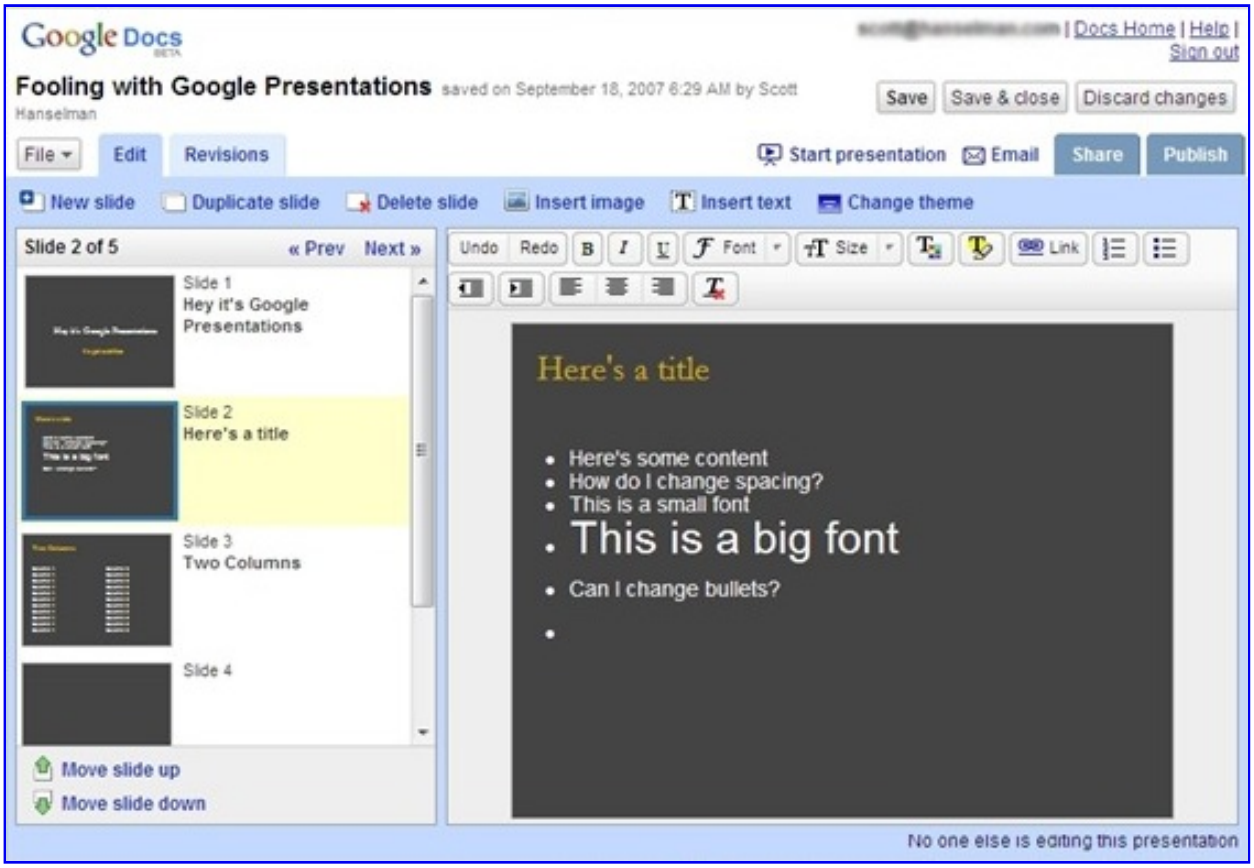
presenter view of the slideshow on a screen (with comments and slides preview) and the normal view on another media, like a video projector.



2.2. On-line collaborative applications

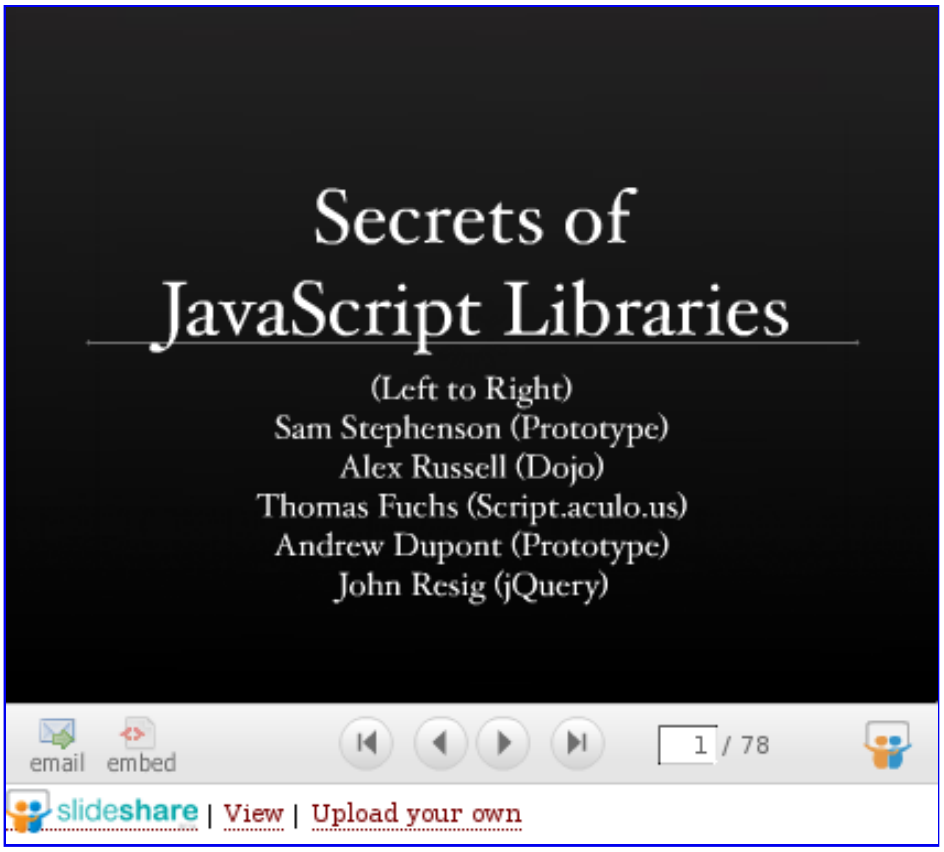
More and more web-based office are offering the possibility to create presentations inside a web browser and to store it online. Their particularity is to make the presentation available from everywhere in the world, providing the user has an internet access and a web browser. It isn't any more required to have a (heavy) application installed locally, nor to take care of software updates.
Most of these applications features a version control and allows many users to edit the document at the same time in a collaborative way and to present it online, but they are not designed to make an offline presentation : there is now way to have two different views on two different medias.
They are appropriated to create basic documents with a limited amount of animations/transitions and types of embeddable elements

(no video, music).



2.3. On-line sharing applications

There is another, less known web-based kind of tool in relation with our project : the presentation sharing platforms. They are used to manage and shares collection of slideshows on the web by producing a presentation applet out of an uploaded slideshow (a kind of Flickr).



2.4. Our approach

Our application can not compete with the number of features of desktop presentation program and our primary goal is not to produce a collaboration tool either.
We have to design an alternative to traditional tools: allowing users, from everywhere, on any platform, to easily create and share visually attractive presentations.

3. Use cases

We've elaborated a set of scenarios corresponding to main tasks users will be able to achieve with our application.
Thanks to them, we are going to describe the composition of the GUI of our application. We are also going to introduce its features, classified in three categories (*must, should and may*) depending of their respective need from an end-user point of view.

3.1. Connection to the application. Academic Point Of

View

Academics access to our application on a subdomain within the intranet of the university (using a dedicated address such as *http://**academics.netshows.**tees.ac.uk*). Our application will provide its own login page despite the fact that it will share its user database with other intranet applications.

3.1.1. login page

must

- provide an explanation about the application
- allow academics to log in the application

may

- display a link to a screencast

3.2. Presentations management.

Once connected, user access the main interface. It is composed of two panels : a treeview of user's presentation and the main panel with a *welcome tab*.

3.2.1. Presentation treeview

The treeview is comparable to a basic treeview of a file system in a file explorer.

must

- expand/collapse (sub)folders as well as presentations
- drag'n drop presentation/subfolder from a folder to another
- right-click on any item to rename it
- right-click on any item to delete it (actually send it to the recycling bin)
- right-click inside a folder to create a new folder or a new presentation (see presentation creation scenario)

- simple click on a presentation to have its description displayed in a part of the welcome tab
- double click on a presentation/slide to open in a new editor tab (see presentation edition)
- right-click inside a presentation to insert a new slide
- drag'n drop a slide from a presentation to another
- copy/paste a slide from a presentation to another
- clone/paste a slide from a presentation to another (a clone is always identical to the original slide)
- right-click on the recycling bin to empty it
- right-click on any item in the recycling bin to definitely delete it

should

- enter key words in a search-field to filter the displayed presentations
- use keyboard to perform previously mentioned actions
- select multiple items to perform previously mentioned actions

may

- use multiple tags instead of folder to classify and filter presentations

3.2.2. welcome tab

must

- Display a welcome message :)
- Inform the user about what he can do with the application

should

- display a tip of the day
- display a link to a FAQ list

may

- display a link to a screencast

3.3. Presentation creation

the user chooses a template and a default layout, then he/she accesses to the presentation edition with a first empty slide.

3.4. Presentation edition

After double-clicking on a presentation or a slide, the treeview is replaced by a vertical thumbnail list of the slides; a new editor tab is opened in the main panel to edit the presentation. If it's a slide that has been opened, then it is displayed inside the tab, otherwise the first slide of the presentation is displayed.

3.4.1. thumbnail list

must

- update the thumbnail as the user edit the slide in the editor tab
- allow all the action possible inside a presentation in the treeview
- right-click to import a set of slides from another presentation

3.4.2. editor tab

To create a slide, the user choose the layout of the slide (by default: one title, one column). Then he edits the default text included in the template - using the *text toolbar* -; add elements (tables, images, shapes, ...) to arbitrary position in the slide - using the *insert toolbar* -; choose transition after which the slide appears and animations of the different elements - using the toolbar on the right.

must

- use the edit toolbar to edit text with features of a basic desktop text editor such as WordPad
- use the insert toolbar and the slide itself to insert and modify depth of elements such as text, tables, images, charts
- use the insert toolbar to insert mathematical expression either in LaTex or MathML, converted to an image
- use the toolbar on the right to modify the layout of the slide

- use the toolbar on the right to choose a transition for the slide
- use the toolbar on the right to add animations to element (and choose whether it will start after a delay or a click)
- undo/redo every modification of the slide
- hide/show the toolbar on the right
- resize/move elements directly on the slide

should

- allow to edit a master slide which define the global style and layout.
- allow insertion and manipulation of other element such as audio file, video, applets, quizzes, …
- switch back to a previous revision of the element/slide /presentation.

may

- allow to rotate elements

3.5. Publication

Once finished, the presentation may be made available to public or downloaded directly from the treeview.

must

- right click on a presentation and check *available to students* button
- right click on a presentation and download it as a standalone HTML presentation

may

- right click on a presentation and generate few lines of html code to embed the presentation within a web page (like a youtube video)
- right click on a presentation and check *available to unregistered users* button

3.6. Connection to the application. Public Point Of View

Students access to our application on a subdomain within the intranet of the university (through an address such as *http://netshows.tees.ac.uk*). There is also a login page for students in order to know which course they can access.

3.6.1. login page

must

- provide an explanation of the application
- allow students to log in the application

may

- display a link to a screencast
- allow unregistered users to access to some parts of some presentations

3.7. Presentations browsing

Once connected, students access to the kiosk. It looks like the main page of academics: a treeview of all presentation, sorted by course, on which user has *read right* (he takes part of the course) in the left panel; and a main panel with a welcome tab.

3.7.1. Treeview

must

- double click on a presentation to preview it in a new tab
- right click on a presentation and download it as a standalone application

4. Project plan

4.1. Actors

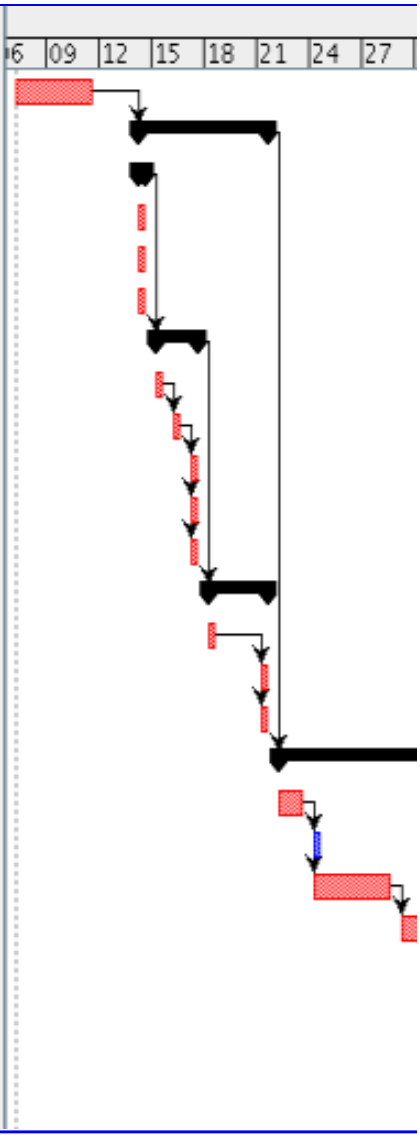
We tried to distribute according to each one's skills and personal background.

- Louis-Rémi BABE is the project leader. He already has some experience with a php framwork *à la* Rails and javascript libraries.
- Clément GONNET is in charge on the User Interface part of the application. He has always been sensitive to aesthetic and usability in the applications he uses and tries to reproduce the best practices in his projects.
- Samuel GARNIER is in charge of the initial documentation. Its training learned him how to be quickly effective with new languages he encounters. He will work mainly on the server side.

We all need to know what other are doing and the basics of the technology they use in order to cooperate.

4.2. Planning

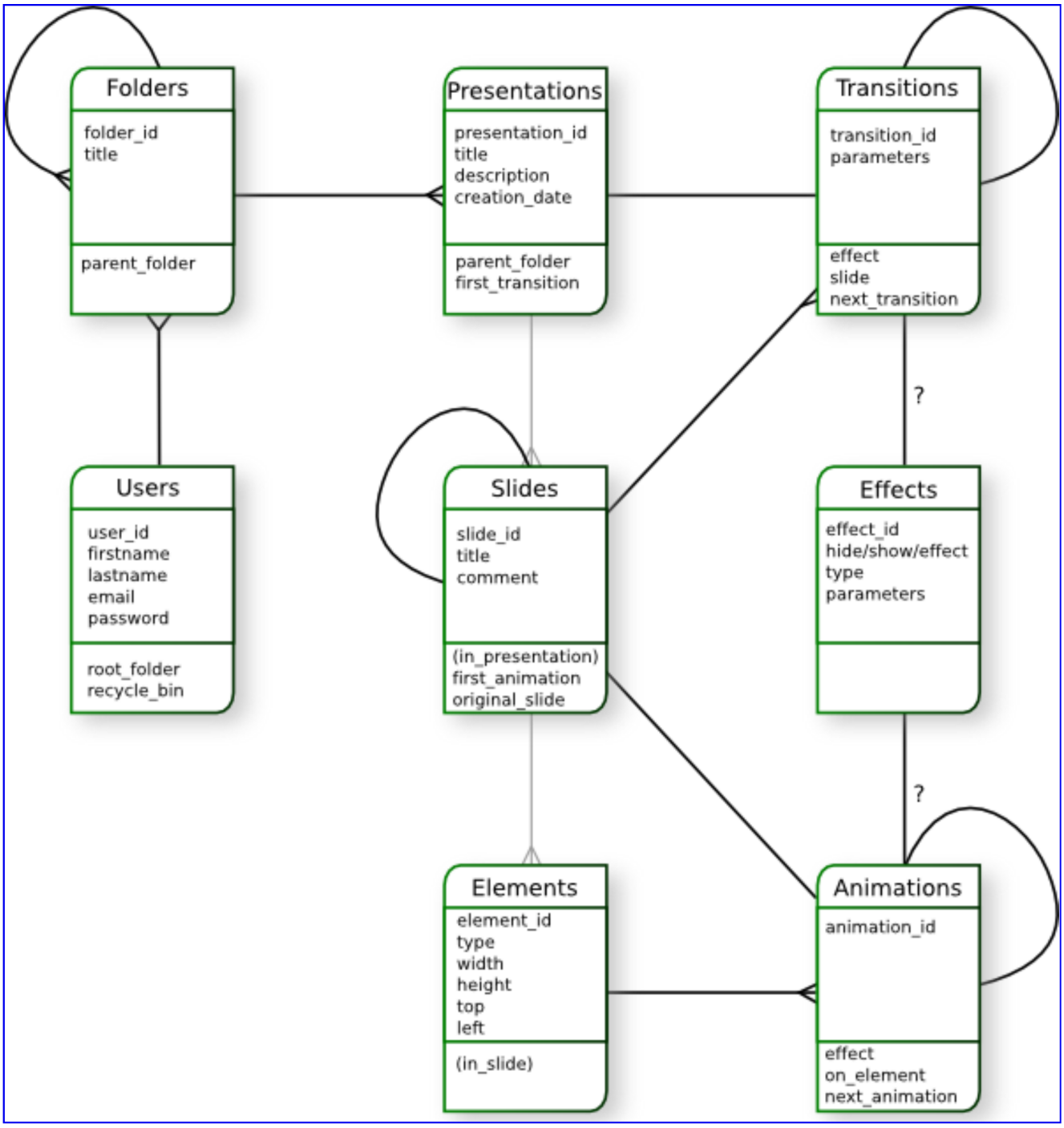
	Nom	Durée	Début	Fin
1	Documentation	5 jours	07/04/08...	11/04/08.
2	**⊟Project Specification**	**6 jours**	**14/04/08...**	**21/04/08.**
3	**⊟Preliminary study**	**1 jour**	**14/04/08...**	**14/04/08.**
4	What? When? Why?	1 jour	14/04/08...	14/04/08.
5	State of the Art	1 jour	14/04/08...	14/04/08.
6	Features	1 jour	14/04/08...	14/04/08.
7	**⊟Detailed Analysis**	**3 jours**	**15/04/08...**	**17/04/08.**
8	Features	1 jour	15/04/08...	15/04/08.
9	Planning	1 jour	16/04/08...	16/04/08.
10	Budget	1 jour	17/04/08...	17/04/08.
11	Actors	1 jour	17/04/08...	17/04/08.
12	Timeline	1 jour	17/04/08...	17/04/08.
13	**⊟Design**	**2 jours**	**18/04/08...**	**21/04/08.**
14	structure	1 jour	18/04/08...	18/04/08.
15	Drafts	1 jour	21/04/08...	21/04/08.
16	Diagrams	1 jour	21/04/08...	21/04/08.
17	**⊟Development**	**67 jours**	**22/04/08...**	**23/07/08.**
18	Pages	2 jours	22/04/08...	23/04/08.
19	Links	1 jour	24/04/08...	24/04/08.
20	design	3 jours	24/04/08...	28/04/08.
21	Features	57 jours	29/04/08...	16/07/08.
22	Test	5 jours	17/07/08...	23/07/08.
23	**⊟Guide**	**5 jours**	**24/07/08...**	**30/07/08.**
24	User guide	5 jours	24/07/08...	30/07/08.
25	Maintenance	5 jours	24/07/08...	30/07/08.



5. Conception drafts

5.1. Data model

Besides the GUI drafts, we have also started to design the data model. Despite the fact that it has been designed to be *future-proof,* it can still grow up during our iterative development process.



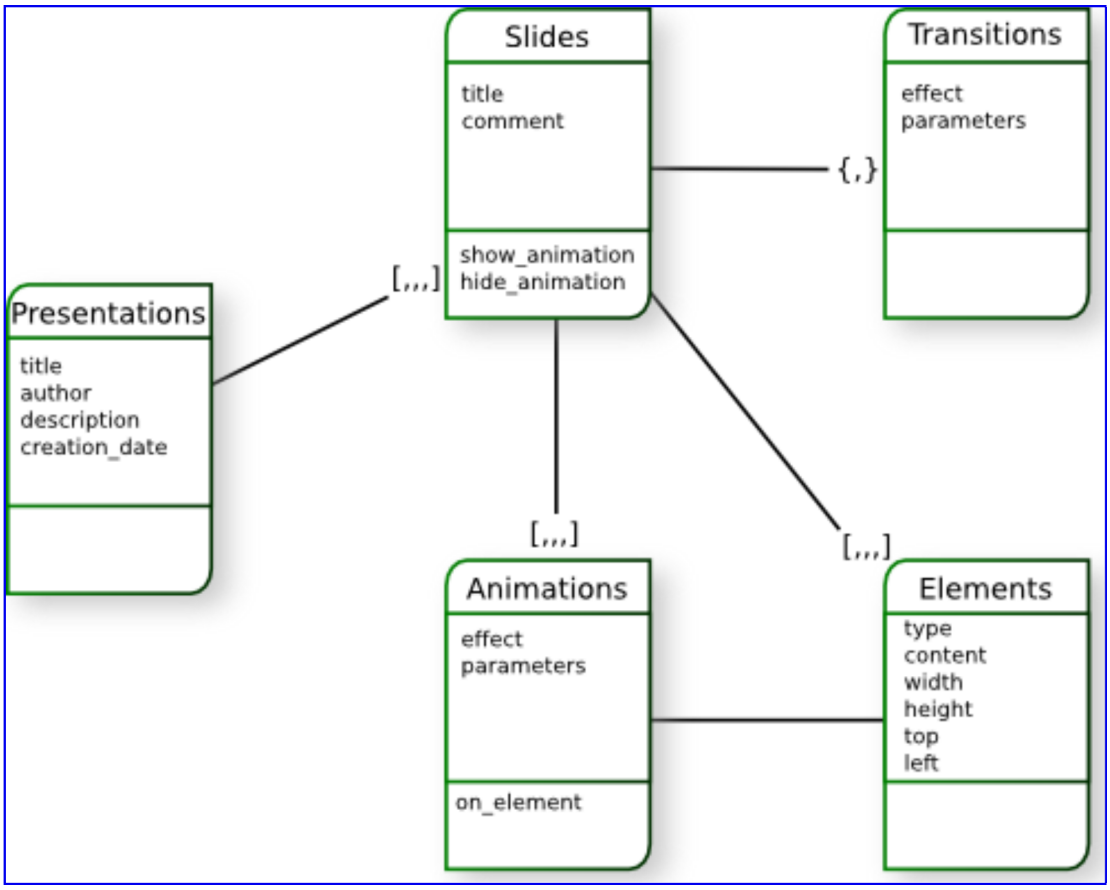
- A user has a root folder and a recycle-bin
- Each folder has a parent folder, except for the root and the bin.

  This kind of link involve a costly (O^n) recursive algorithm to delete a folder but the addition and relocation of a folder is the easiest. The listing of a folder is not a problem thanks to the use of a "lazy treeview".
- A presentation is in a folder and is linked to its first transition.

- A presentation is a singly-linked list of transition showing or hiding a slide.
  This choice was made (instead of a linked-list of slides which seem more natural) because it appears that a transition is not part of a slide: you should be able to clone a slide and have it displayed by two different transitions in two different presentations.
- A transition can correspond to an effect (when the user add a slide to a presentation, a transition is added at the end of the list and a linked to a new slide, but has no graphical effect by default).
- If a slide is linked to another slide (by its original_slide attribute), it means that this slide is a clone and he would behave as a *symbolic link*. A slide has one or two transition and is linked to its first animation.
  For performance purposes, it has been decided to link each slide to its presentation (which should speed up a presentation deletion for example).
- A slide is a singly-linked list of animation hiding, showing or applying an effect to an element (moving, bouncing, ...).
- An animation can correspond to an effect.
- An element can have many animations.
  For performance purposes, it has been decided to link each element to its slide.

5.2. Data model of a standalone presentation (exported in JSON)

When a user want to download a presentation as a standalone HTML file, we have to extract all the relevant data from the data base, and to slightly simplify the data model to output a presentation structure as JSON.
HTML could have probably been used instead of this JSON object, however, not only json is less expensive for our bandwidth, but it is also easier to generate. A small part of the processing is delegated to the client side which will create the DOM elements out of it.



5.2.1. sample

```
var presentation = {
    'title' = "",
    'author' = "",
    'description' = "",
    'creation_date' = "",
    'slide' = [
        {
            'title' = "",
            'comment' = ""
            'show_animation' = {
                'effect' = "",
                'parameters' = {...}
            },
            'hide_animation' = {
                'effect' = "",
                'parameters' = {...}
            },
            'animation' = [
                {
                    'effect' = "",
```

```
                    'parameters' = {...}
                },
                ...
            ],
            'element' = [
                {
                    'type' = "",
                    'content' = "",
                    'width' = "",
                    'height' = "",
                    'top' = "",
                    'left' = "",
                },
                ...
            ]
        },
        ...
    ]
}
```