



João Francisco Silva Caeiro dos Santos

Mestrado em Engenharia Informática

Relational Navigation and Archiving of Multimedia Information for Contemporary Dance

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientador : Prof. Doutor Nuno Manuel Robalo Correia,
Professor Catedrático, Faculdade de Ciências e
Tecnologia da Universidade Nova de Lisboa

Júri:

Presidente: Prof. Doutor José Legatheaux

Arguente: Prof. Doutora Teresa Chambel

Vogal: Prof. Doutor Nuno Correia



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2012

Relational Navigation and Archiving of Multimedia Information for Contemporary Dance

Copyright © João Francisco Silva Caeiro dos Santos, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

A Faculdade de Ciências e Tecnologia e a Universidade Nova de Lisboa têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

For my family and friends.

Acknowledgements

To my thesis advisor Prof. Nuno Correia, for the opportunity to work in this project, for all his guidance, the support and confidence in my work as well as his availability throughout the last year, all of which was essential for the successful elaboration of this work.

To Prof. Carla Fernandes, for all the support given throughout the elaboration of this work and the opportunity to work alongside her in the TKB project. Also for her availability and remarkable feedback, without which this work wouldn't have gone so far.

To all the persons in the TKB project, in particular Urândia Aragão for her amazing ideas, hard work and patience in our brainstorming sessions, and Evi Dimakopoulou for her work done in the TKB website and support throughout the development process.

To my faculty, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, which has been my second home and a big part of my life for the past years, for all the knowledge and academic experience it has provided me.

To the TKB project- A Transmedia Knowledge Base for contemporary dance (PTDC/EAT-AVP/098220/2008), in which this thesis is inserted, for granting me a scholarship which helped support my studies for the past months.

To all my colleagues, with whom I had the pleasure of sharing the last years in this Faculty, for all the friendship and support, which helped lighten the hard work done throughout this course.

To my friends in general for making my life complete, helping me keep moving forward and brightening all my days. To Mihalina Georgieva and all the wonderful people I've met which have helped to make these past years unforgettable. To André Ascensão for all the motivation, support and help throughout this work. To Pedro Dias and Ana Sofia Canelas for always being there when I needed and for all the support and encouragement given.

A special thank you to my mother Maria João da Silva and father Vitor Santos for loving and supporting me through all the decisions in my life. To my sister Raquel Santos for loving me, supporting and taking care of me for as long as I can remember. And to everyone else in my amazing family for all the love and support through these years.

Abstract

Technology continues to evolve at an incredible rate and with it, the number of people adhering to new digital trends. As the production of multimedia content such as photographs and videos becomes accessible to more people every year, so does the amount of digital content increase exponentially. Consequently, it becomes a hard task to create systems in order to provide efficient storing and browsing of multimedia content. Multimedia Web archives are one of the most popular solutions found for these issues. By providing organized and connected information storage with efficient browsing and social networking features, these systems become the main platforms used to store and share photographs and videos in the internet.

This work is done in the scope of the TKB project: Transmedia Knowledge Base for Contemporary Dance and the goal of this thesis is to develop a system for multimedia information storage and relational content navigation. The analysis of multimedia archiving systems done throughout this thesis extends to those specific for Contemporary dance as it is one of the main focus of the work. The contents which will be integrated in the archive include typical multimedia information such as images and videos, as well as annotated videos exported from specific platforms. Connecting all the information within the archive through taxonomy and content hierarchy allows the definition of the intended relational approach. Setting connections between content and users allow the creation of graphs, which will serve as a basis for all the browsing, navigation and searching done throughout the system.

Keywords: multimedia archive, relational navigation, contemporary dance, web archive

Resumo

A tecnologia continua a evoluir a um ritmo incrível e com ela, o número de pessoas que aderem a novas tendências digitais. À medida que a produção de conteúdos multimédia tais como fotografias e vídeos se tornam acessíveis a mais pessoas todos os anos, também a quantidade de conteúdos digitais aumenta. Consequentemente, torna-se uma tarefa difícil criar sistemas que forneçam o armazenamento e navegação eficiente de conteúdos multimédia. Os arquivos Web de informação multimédia são uma das soluções mais procuradas para resolver estes problemas. Ao fornecerem o armazenamento de informação interligada e organizada, com componentes de pesquisa e interacção social eficientes, estes sistemas tornam-se uma das principais plataformas usadas para armazenar e partilhar fotografias e vídeos na internet.

Este trabalho está inserido no âmbito do projecto TKB: Transmedia Knowledge Base for Contemporary Dance e o objectivo desta tese é desenvolver um sistema para armazenamento e navegação relacional de informação multimédia. A análise de sistemas de arquivo multimédia feita ao longo desta tese estende-se a arquivos de dança contemporânea visto ser um dos focos principais deste trabalho. Os conteúdos que serão integrados no arquivo incluem ficheiros multimédia comuns tais como imagens e vídeos, bem como vídeos com anotações exportados de plataformas específicas. Ao interligar toda a informação dentro do arquivo através de taxonomias e hierarquia de conteúdos torna-se possível definir a abordagem relacional pretendida. A definição de ligações entre conteúdos e utilizadores permite a criação de grafos que serviram como base para toda a navegação e pesquisa feita no sistema.

Palavras-chave: arquivo multimédia, navegação relacional, dança contemporânea, arquivo web

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Problem context, definition and objectives	2
1.3	Main contributions	4
1.4	Document organization	4
2	Related work	5
2.1	Multimedia archives	5
2.1.1	Typical web archives	6
2.1.2	Dance archives	9
2.2	Relational navigation of information	14
2.2.1	Digital Vaults	16
2.2.2	Dancers Project	17
2.2.3	Bestiario: Videoshpere and Bestiary	18
2.3	Timelines and video annotation visualization	21
2.3.1	ELAN	22
2.3.2	Creation Tool	24
2.3.3	Synchronous Objects	26
2.4	Summary	27
3	Archive and relational navigation	29
3.1	Project context	29
3.2	Archive structure and contents	30
3.2.1	Content types	30
3.2.2	Structure and organization	34
3.3	Relational navigation and search	37
3.3.1	Archive navigation	37
3.3.2	Individual archives	38
3.3.3	Search and related contents	39

3.4	Summary	42
4	Features details and implementation	43
4.1	System implementation decisions	44
4.2	Technology	46
4.2.1	Drupal CMS	46
4.2.2	HTML5	49
4.2.3	Data-Driven Documents - d3.js	50
4.3	Backoffice	52
4.3.1	Drupal backoffice and content creation	53
4.3.2	ELAN and Creation Tool annotations parsing and storage	56
4.4	Content hierarchy and relational navigation	58
4.4.1	Content hierarchy	58
4.4.2	Graph representation	62
4.4.3	Search	67
4.5	Annotations and media content visualization	68
4.5.1	Multimedia content types	68
4.5.2	Annotated content	70
4.6	Summary	74
5	Evaluation	75
5.1	Results analysis	76
5.1.1	Users	76
5.1.2	Navigation	76
5.1.3	Searching	77
5.1.4	Content creation	78
5.2	Summary	79
6	Conclusions and future work	81
6.1	Challenges and system limitations	82
6.2	Future work	82
6.2.1	Content preview in graph navigation	83
6.2.2	ELAN annotations searching mechanism	83
6.2.3	Video transcoding	83
A	Appendix: Questionnaire	89
A.1	Questions	89
A.2	Results	92

List of Figures

2.1	Flickr most popular tags	7
2.2	Annotation on a photograph	7
2.3	Vimeo’s categories and content organization	9
2.4	Siobhan Davies: content and related media	10
2.5	Siobhan Davies: Kitchen media	11
2.6	Digital Dance Archives: Visual Search	12
2.7	Digital Dance Archives: Search Results and manipulation	12
2.8	Jacob’s Pillow Dance Interactive: Category results	13
2.9	Jacob’s Pillow Dance Interactive: Quiz	14
2.10	Digital Vaults: Related items	16
2.11	Dancers: Relational Navigator	17
2.12	Videosphere: outside view	19
2.13	Videosphere: inside view	20
2.14	Bestiario Bestiary	21
2.15	ELAN: Overall interface	23
2.16	ELAN: Multiple tier view	24
2.17	Creation Tool	25
2.18	Synchronous Objects	26
3.1	Gallery Template	31
3.2	Creation Tool Template	33
3.3	Elan Template	34
3.4	D3 hierarchical layouts	35
3.5	Edit annotations: ELAN	36
3.6	Global archive	38
3.7	Individual archive navigation	39
3.8	Tag and Keyword Search - Table	40
3.9	Related Content - Graph	41

4.1	Drupal Technologies	47
4.2	Drupal Hook System	48
4.3	HTML5 Video formats	50
4.4	D3 hierarchical layouts	51
4.5	Backoffice Area	54
4.6	Backoffice Menus	55
4.7	Annotation database tables	57
4.8	Annotation controls	58
4.9	Menu items	59
4.10	Tag creation	61
4.11	Archive context menu	68
4.12	Annotation information structure	71
5.1	Results of the evaluation of the main archive's structure and navigation. .	77
5.2	Overall annotation filtering and visualization evaluation	78
5.3	Newly created content	79
5.4	Average system ratings	80

List of Tables

4.1 CMSs comparison	45
-------------------------------	----

Listings

4.1	Node creation	51
4.2	Accordion configuration	55
4.3	Custom menu creation	60
4.4	Main graph creation	63
4.5	Graph node filtering	64
4.6	Custom menu data	65
4.7	Forced tree layout	66
4.8	Click function	66
4.9	Mergin search input	67
4.10	Slideshow markup	70
4.11	Annotation document	71
4.12	Toggle video size	72
4.13	Timeline scroll handler	73
4.14	Canvas paint annotation	74



Introduction

Each day millions of people store their personal photos and videos online. Either by using a personal website, blog, social network or multimedia sharing application, all this content will be available to be seen and searched through in some way. With the amount of multimedia information online increasing at an unprecedented rate, it becomes a hard task to store it, in order to avoid it becoming irrelevant as opposed to a good source of organized information. The creation of this kind of multimedia platforms is a complex task, which obviously is not at the reach of every common user of the web. Therefore, it becomes essential the increase of such publicly accessible multimedia systems, as it happened in the past years.

Multimedia archives in general have become a popular choice in order to solve multimedia organization issues. Large companies such as Yahoo ¹ and Google ² have developed this type of platforms, providing multimedia storing and sharing services for global public access. However, when it comes to storing large and searchable quantities of multimedia content originated from different sources, the challenge becomes the design of efficient displaying and browsing techniques. In other other words, gathering all the related content into an effective multimedia navigational system.

1.1 Motivation

Most of the existing online platforms which allow multimedia storage focus on organizing the content in the same way, having a low amount of perceived relations between

¹<http://www.yahoo.com/>

²<http://www.google.com/>

them. It is common to find multimedia content organized by author or owner, according to a pre defined category, as well as tagged with specific words chosen by the owner of the content, forming the popular tag-cloud we can find almost everywhere. Providing the possibility to store different multimedia types (e.g., video, audio, images), and establishing relationships between them, in such way that allows this content to be seen in a more global context, can be a good way to extend information navigation. With this, someone browsing for a specific category or subject does not have to be confined to such a limited universe and could be led to different, yet still relevant information.

Another important aspect of online multimedia storing is that, despite the fact that two pieces of content fall into the same category or are related in some way, this link between the two will be unnoticeable if they do not share the same type as well. Taking, for example, pictures and videos both inserted in the category "Contemporary Art", it is possible to search for this content online through search engines which will redirect the user to its source, separately on multimedia sharing applications for both image (e.g., Flickr, ³, Picasa ⁴) and video (e.g., YouTube ⁵, Vimeo ⁶), or on a personal web page or social network in order to combine the two.

Even though these are valid solutions for most of the cases, sometimes it is preferred or even required to have a broader view of a certain subject or topic. Having the possibility to display different types of content and the way they relate to each other is a natural way of representing an organized structure as well as the overall context of the information we want to store.

1.2 Problem context, definition and objectives

This work is done in the scope of the project TKB – Transmedia Knowledge Base for contemporary dance. The TKB project, developed together with Faculdade de Ciências Sociais Humanas – FCSH/UNL, aims at the design and development of a multimodal knowledge base to document, annotate and support the creation of contemporary dance pieces and rehearsals. This transdisciplinary project has the goal of providing a research space for critical exploration of the relationship between linguistics, dance, new digital media and thought/ consciousness. The project consists of several components developed previously:

- Linguistic annotation- dance rehearsal video annotation of specific terminological glossary of choreographic elements for the construction of a Portuguese contemporary dance archive;

³<http://www.flickr.com/>

⁴<https://picasaweb.google.com/>

⁵<http://www.youtube.com/>

⁶<http://vimeo.com/>

- Software development for annotation and motion analysis of human body in dance videos;
- Creation-oriented tool [19, 31] – tool to support the choreographer’s creative process as well as the individual archive manipulation.

This work aims for the development of an archiving and navigation system on relational multimedia information, including image, audio, video and annotated videos. The goal of this work is to support the content generated through creative processes, such as dance rehearsals and pieces, in order to create a navigational multimedia network of this material and thus, the knowledge base. Such material may have different sources and types. Considering that multimedia content is the main focus of the archive, it is helpful to separate it into two categories:

1. **"Pure" multimedia content:** picture, video and audio which do not require any additional support material other than text (title, description, etc.);
2. **Annotated videos:** material generated by the annotation software ELAN [6, 17] and Creation Tool [19, 31] (developed in the same project).

As the main archive supports content from multiple users, the respective content items are to be divided and organized according to its owner. Therefore, the global archive is separated into smaller, individual archives for each user, whose structure should be defined by each creator, which would consequently define how the relational navigation is done within the archive, having always the content as the foundation level. As one of the goals is to provide an effective navigation along the archive, the relations between either each section, page or content are represented in a tree or graph-like fashion. Considering this principle, the main objectives of this work can be described in the following way:

- Individual and customizable archive which allows the user to present his work through the storage of multimedia content;
- Relational navigation within the global archive and between individual archives;
- Multimedia content visualization of both simple and annotated materials.

In order to accomplish these goals, it is essential to take into consideration certain implementation features. Firstly, storing and managing the content should be treated as a priority as it is crucial to maintain stability and security when both the total amount and individual size of the multimedia content increases significantly. Then, considering the relational navigation between content and individual archives, it is necessary to achieve a solution where the amount of information displayed is maximized, without compromising the overall usability and legibility of the content, thus avoiding an information overload situation. The same applies to the visualization of annotated video content. Considering that the amount of annotations in a single time frame can be considerably

large, it becomes crucial to devise ways of properly displaying and distinguishing them. Therefore, the work presented in this thesis proposes an approach for the implementation of the multimedia archive and the described features, along with a solution for the aforementioned issues.

1.3 Main contributions

The main contributions resulting from the development of this work are:

- **Online platform for information archiving** - Development of an online multimedia archiving platform providing users the documentation of their work in the creation of their own personalizable archive;
- **Annotation visualization** - Being one of the main types of material to be stored in the archive, the complex visualization of the annotations in the videos will be one of the main features.

1.4 Document organization

The following represents the structure of the remaining document:

- **Chapter 2: Related Work** - contains an overview of all the relevant topics related to the development of this work, addressing multimedia web archives, both general and specific to contemporary dance. Follows by a review of the use of relational navigation in information browsing, ending with an analysis of various annotation visualization applications and corresponding implementations.
- **Chapter 3: Archive and Relational Navigation** - overview of project context, followed by the overall archive structure and content types supported, as well as the details of the main functionalities implemented, including the relational content navigation and searching.
- **Chapter 4: Features Details and Implementation** - discussion of the various implementation and technology decisions, analysis of the implementation of the various components of the system, including core system and backoffice, relational navigation and multimedia content visualization.
- **Chapter 5: Evaluation** - presentation and analysis of the results obtained through the evaluation process.
- **Chapter 6: Conclusions and Future Work** - overview and conclusions regarding the work done as well as future work directions.



Related work

With the increasing amount of information being generated every day by millions of people, it becomes essential to design ways of better organizing this content. Even though we want to have all the content available, it can be hard to navigate through all of it, specially when there is no kind of connection or hierarchy between the content items.

For the development of a multimedia archive it is necessary to properly understand how multimedia information should be organized, in order to provide a better, more natural and appealing way of navigating through the archive and browsing its content. With this purpose, it became essential to analyse existing systems related with the main focus of this work. Since the archive can be a truly complex system as a whole, it is better to separate the context of the analysis in three separated parts. First, as it is the main topic, multimedia archives and galleries will be detailed. In the next section the focus will be on content navigation through graph or relational support. Finally, as one of the main content types of this archive will be annotated videos, it was necessary to study video annotation systems and how these annotations are shown on a linear timeline.

2.1 Multimedia archives

The amount of multimedia content available online increases daily at a fastening rate. As such, digital libraries and archives are built in order to organize this content, allowing users to browse through the collection.

With the large variety of personal web archives available, it was necessary to separate them into two distinct categories. In the first category we have the typical web media

archives, in which we can find platforms such as Flickr ¹ or YouTube ². In the second category we fit the contemporary dance archives, which are part of the main focus of this thesis. In the following sections more will be detailed on these different categories, how they are typically built and in what way can that implementation be used for the purposes of this work.

2.1.1 Typical web archives

The introduction of digital photography changed the general consumer's behaviour when taking a picture. During the past years the prices of digital cameras have dropped drastically, making photography affordable to a wider range of people. Moreover, almost every recent mobile phone has a built-in camera with decent quality. This allied to the high capacity of current memory cards, and the increasing use of high-speed internet enables people to store great amounts of digital photographs. As a result, providing decent platforms for the storage and browsing of this content has become essential.

Most of the media content available on the Web have very few to no information associated to it, sometimes being limited to a small description or a collection of tag words. This being the way that the content is stored, it is in a similar way that this content is displayed to a user. Independent of being a video, image or mixed archive, there are normally pre-defined ways of organizing the added content. It can be sorted by the date it was uploaded, by its popularity, divided into different albums or separated according to filters, such as tags applied to each one.

Even though most Web archives provide ways of categorizing and eventually organizing multimedia content, it is not common to find those which enable relational navigation nor the browsing of multiple content types in the same visual context. The following section describe some of the most popular web archive systems, both for picture and video.

2.1.1.1 Flickr

Flickr ³ is one of the most popular online photo management and sharing platforms, allowing users to easily store, search through and share their photo collections. Working mainly as a social network where users can upload and share their personal photographs, it is also widely used by bloggers in order to host images which they want to embed in blogs, websites and social media. It is not necessary to be a registered user to access most of the photos on the website although it is required to create an account in order to upload any content and interact with other users.

¹<http://www.flickr.com/>

²<http://www.youtube.com/>

³<http://www.flickr.com/>

Users have the possibility to comment and add descriptions to every image on the website. It is also possible to create groups and tag photos that relate to it, meaning that any user can add images to groups which categories seem relevant. This gives a dynamic life span to all the content, where old images can remain "active" for longer, contrary to what is found in most blogs and archives. All the tags are used for organization of the content as well as for its browsing, generating different visualization contexts such as the tag cloud in figure 2.1.

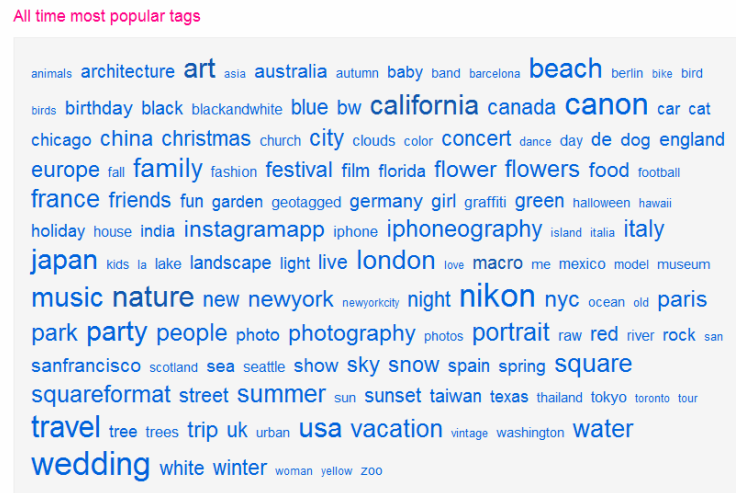


Figure 2.1: Flickr most popular tags

This tag cloud lists groups of photos with the most commonly used tag words, providing as well a service to search through content by tag terms. Another feature available is the annotation of images, seen on figure 2.2, which is a way the creators of Flickr found of capturing the conversation and ideas about a picture.

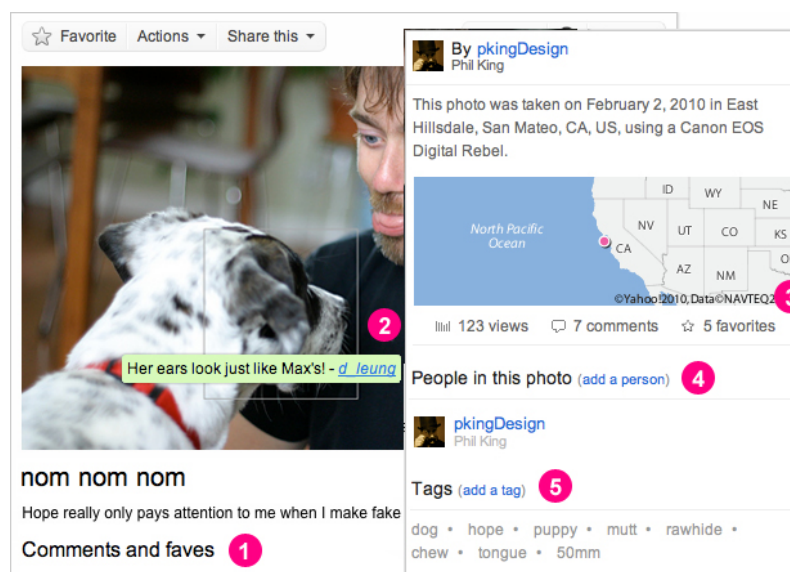


Figure 2.2: Annotation on a photograph

Flickr has proved to be one of the most innovative photo sharing platforms available. Providing a relevant set of features which make it easier to upload and share photographs, as well as the capability of integration and customization through a free-to-use API, resulting in the most varied custom applications, all with access to the enormous amount of data on their servers.

2.1.1.2 Vimeo

Video sharing platforms work similarly to image ones in the sense that every user can access the publicly available content, and only registered users can upload and mark favourite videos, leave comments and share with other users. All of these work somehow similarly to YouTube ⁴, where the capability to connect with social networks is a must-have feature and where it is possible to find videos of every type, related again by tag words and global categories. In this way we get a mixture of interesting and actually relevant content, with other that may not be of use at all, which could appear in the same search or browsing context just because of similar tags or descriptions. It is also normal to notice that, as the quantity of content available increases, its overall quality and relevance tend to diminish. Therefore, considering that one of the main focus of this work is to provide an organized knowledge base with natural navigation and quality content, it seems relevant to analyse as well one of the video sharing web platforms with the most quality content.

Vimeo ⁵ is a video sharing website with a large community of video enthusiasts, artists, indie film makers and has gained the reputation of having high quality content. Among the users it is possible to find music artists who choose to release their music videos or entities or networks who want to post high quality broadcasts, due to the support for high definition video encoding and reproduction not easily found elsewhere. Also, in order to maintain the quality of their content there are users assigned to moderate uploaded content and generally they follow a few guidelines:

- No inappropriate content;
- No gaming videos, unless they follow a unique script;
- Video must have been created by the uploader.

Regarding content organization and navigation, the same model is used as in most platforms, placing videos in the correspondent categories. When a user uploads a video he can add to it related keywords and manually insert it previously created groups, which will further on establish the relation between all the stored videos and thus improving

⁴<http://www.youtube.com/>

⁵<http://vimeo.com/>

related content browsing. In figure 2.3 we can see how Vimeo displays the various categories (1) with a brief preview of sub-categories, and how the videos are organized and related to each other within one category (2).

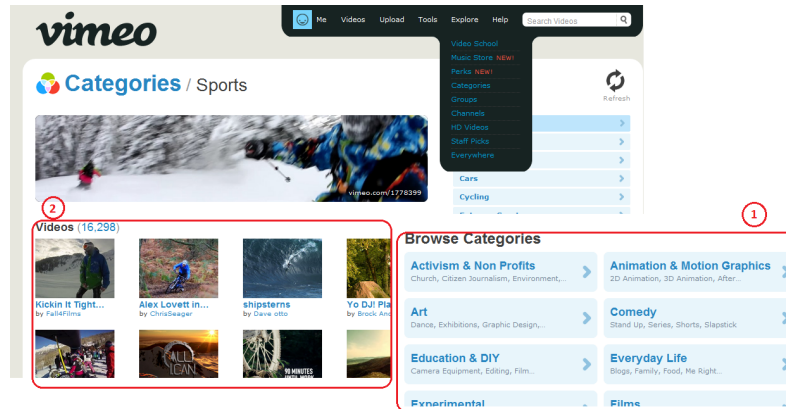


Figure 2.3: Vimeo's categories and content organization

Vimeo also provides support for visualization in various platforms, including mobile smartphones, AppleTV⁶, Boxee⁷, among others. For integration and customization, developers can find their API⁸ documentation which provides an easy way of accessing Vimeo's stored data.

2.1.2 Dance archives

As any other type of archive, the goal of a dance archive is to store a collection of historical records either in physical or, as with this work, a digital location. It consists mainly of important documents that have accumulated through the years of a company's activity or an individual's personal work. In the case of dance archives or, more specifically, a choreographer's archive is many times considered any material that relates to their artistic creations, including choreographic notes, rehearsal videos, audio recordings, photographs, among others. Dance archives are usually built with the purpose of cataloguing these different types of information related to one or several works and pieces of a choreographer or dance companies.

The dance archives found throughout the Web vary both on the contents they store as well as the way these are displayed and connected. From those which focus more on the works of only one choreographer [11], to others which collect a large variety of genres, creators and periods [7], passing through archives which focus and analyse extensively a single choreography [13] and even social networks [2].

⁶<http://www.apple.com/appletv/>

⁷<http://www.boxee.tv/>

⁸<http://vimeo.com/api>

In the following sections some of these archives are analysed, with attention to what makes them stand out when compared to others, and how they can help in the development of this work.

2.1.2.1 Siobhan Davies RePlay

Siobhan Davies RePlay⁹ is the project of Siobhan Davies, former dancer and choreographer in the London Contemporary Dance Theatre, as well as the founder of the Siobhan Davies Dance Company. This archive aims to gather all of the materials and documentation associated with Davies' work, both in the context of the company and before its creation. With support from many of Davies' collaborators, who in some cases contributed with items from their private libraries, RePlay became one of the first archives of this type in the United Kingdom. One of the main goals of the creation of this archive is to store and share content online, which otherwise would remain inaccessible and in some cases would never be seen by anyone after the date of its creation.

The collection on this archive contains numerous Siobhan Davies' own dance works, as well as a few related projects. The organization of the content is done in a similar way throughout the whole archive, as can be seen in figure 2.4. Whether it is in the context of a dance work, a person or a media file, there is always a title, descriptive summary, related artists and works 2.4(1) as well as the media related to it 2.4(2).

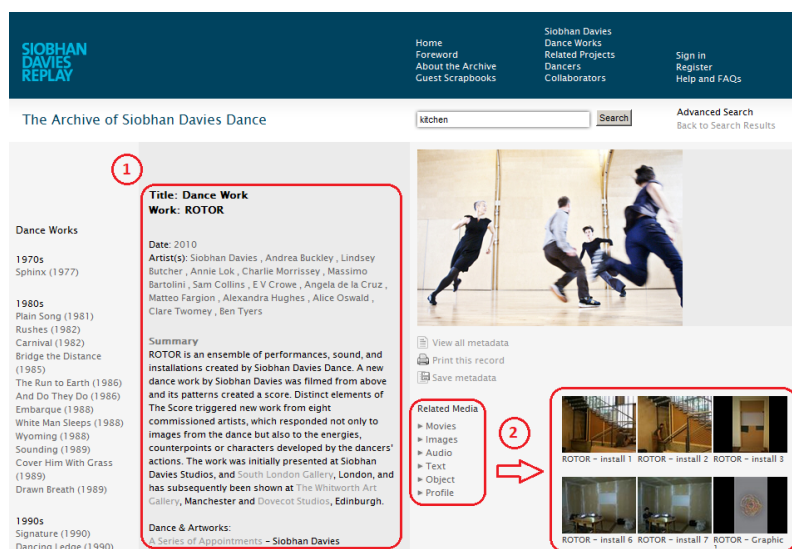


Figure 2.4: Siobhan Davies: content and related media

Featuring a good search interface with simple and advanced modes, both over the entire archive collection, which provides alternatives to the original navigation and browsing of the archive. In addition to these browsing options, it is also worth mentioning two included functionalities, the Scrapbook and the Kitchen. The first provides an easy way

⁹ <http://www.siobhandaviesreplay.com/>

for users to save and organize media found within the archive, both for personal future reference and for sharing with other registered users. The latter is an interactive presentation of works and all the related material used in the process of its creation. This presentation is only implemented in two of the works available within the archive, being one of them, "Bird Song", represented in figure 2.5.

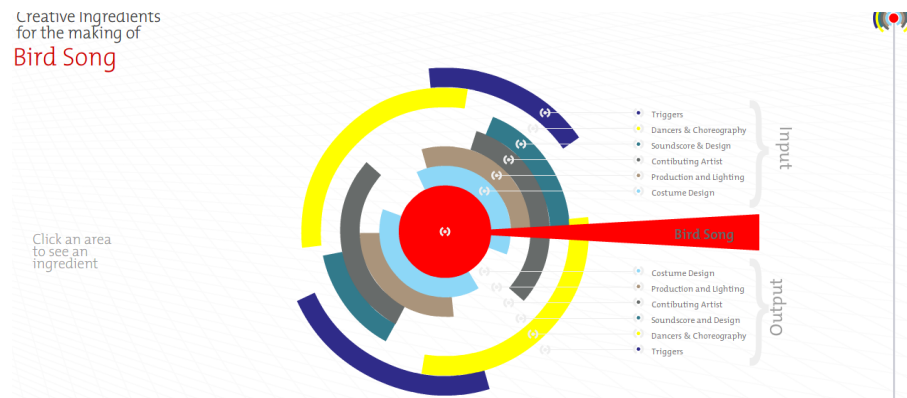


Figure 2.5: Siobhan Davies: Kitchen media

This visualization option aims to gather, in an interactive digital environment, all the concepts, ideas and people which were part of the development part of the final work. The layout of this presentation is divided into several levels. The upper level where the initial, more abstract concepts and ideas reside, to the center, bottom level where we can find the final result of the work, passing through the people involved in it as well as how all of this is related and has contributed to it. The result is a structured and unique browsing interface, which tries to provide an explanatory tour of the creation process of the work in question. Only to note the fact that both the video visualization and the Kitchen interface are both done with resource to Adobe Flash ¹⁰.

2.1.2.2 Digital Dance Archives

From all the dance archives analysed, this is the one in which the browsing experience becomes truly unique. Digital Dance Archives (DDA) ¹¹ gathers material from six different collections, including Siobhan Davies RePlay mentioned earlier. In addition to the conventional text-based searching, DDA has implemented a unique search platform which locates content with similar shapes, colors, gestures and movement patterns. In order to proceed to this kind of search it is necessary to select an object first so, in order to explain more clearly, we have the photograph chosen in figure 2.6 as an example. We start by selecting the area intended for the search in the photograph 2.6(1), and proceed to clicking on one of the icons in 2.6(2) for color, pose or movement search.

¹⁰<http://www.adobe.com/products/flashplayer.html>

¹¹<http://www.dance-archives.ac.uk/>

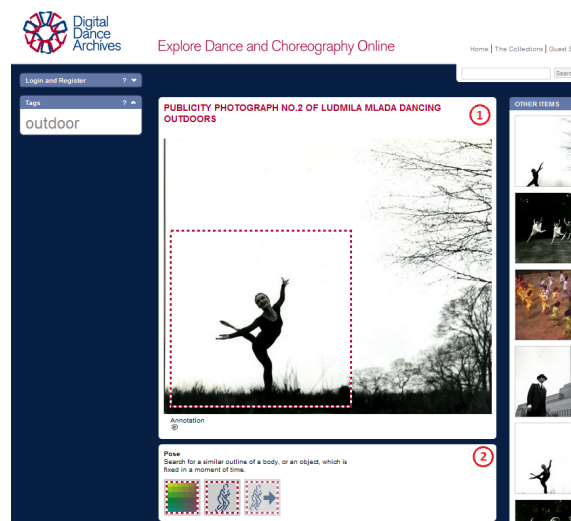


Figure 2.6: Digital Dance Archives: Visual Search

Upon the retrieval of the search results, the items are arranged in a square grid as in figure 2.7, each one with varying dimensions according to the number of items to be displayed. Having the automatic layout of the search results 2.7(1), it is possible to re-arrange each item individually as well as resize them, providing countless variations of visualization options as in 2.7(2). Additionally to the layout customization possibilities, the user can further filter the results selecting the intended collection to show 2.7(3). As a quick search function there is also the possibility to select popular key words from the tag cloud, which is visible at all times.

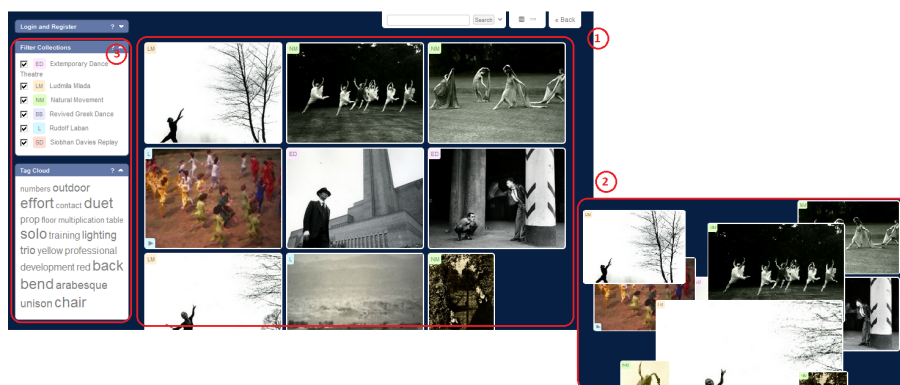


Figure 2.7: Digital Dance Archives: Search Results and manipulation

The quantity and diversity of the content stored in this archive is considerably small, at least when compared to generic media archives such as YouTube or Flickr. This is an aspect where a lot of projects fail. By trying to store the many items into one collection and not focusing primarily on the usability, the result is an archive which may have a good database but fails to deliver the intended navigation experience.

2.1.2.3 Dance Interactive

Jacob's Pillow Dance Interactive¹² is a project which aims to build audiences and appreciation for dance in general and for Jacob's Pillow, one of the most prized dance festivals in the United States of America. Considered "the dance center of the nation" by *The New York Times*, this enormous historic landmark is home to not only the festival, but as well performances, exhibitions and courses. The contents of this archive consist of all the recorded performances in Jacob's Pillow, the oldest dating back to 1936.

Contrary to most of the multimedia archives, in Dance Interactive the relations between each item are not visible. In order to filter and browse through the desired content the user may choose between two different methods. The first one is similar to any other category browsing, where the user starts by choosing an upper level of filtering, between the three available: **Artist**, **Genre** or **Era**. The next step is to select a sub-category. For Artist and Era there are a number of pre defined intervals of both artists names and years, corresponding to an Era (e.g., A-C; 1990-1999). In genre there are as well a few sub-categories, this time separating the media by the dance genre it is part of (eg., Ballet, Contemporary). The results of these two selections are presented in a single row, as in figure 2.8, in either alphabetic or, as in this case, chronological order.

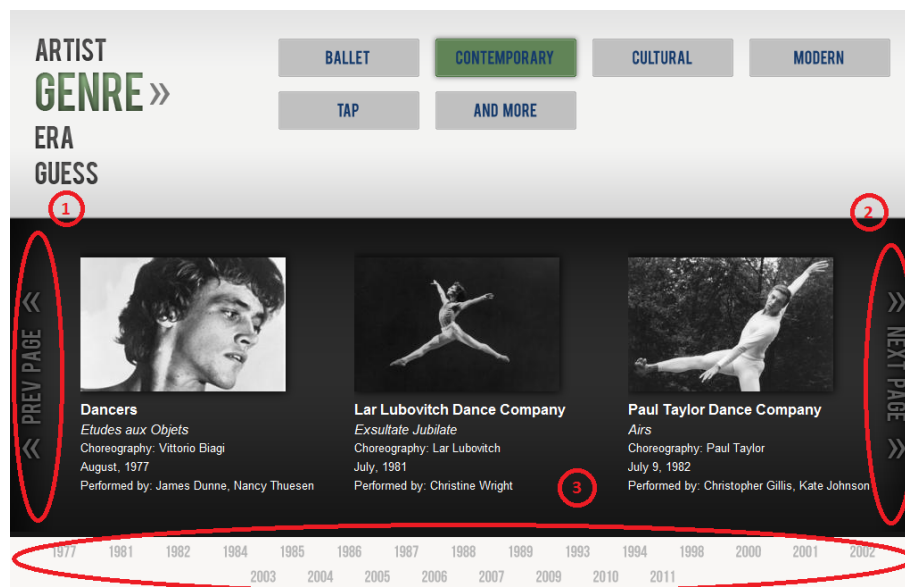


Figure 2.8: Jacob's Pillow Dance Interactive: Category results

It is possible to navigate through the results either back and forward page by page in figure 2.8(1) and 2.8(2), or by jumping to a specific year with 2.8(3). By selecting one of the items in the results, the user can consult detailed information about the performance or artist chosen, as well as see the correspondent video.

¹²<http://danceinteractive.jacobspillow.org/>

The second browsing method of this archive is a rather different approach and consists of a quiz. In this game-like browsing a video is played, accompanied by a question and four possible answers as in figure 2.9. This question is always regarding the choreography or performers in the video. By answering correctly, the user can read further details about this particular item, or proceed to the next question.

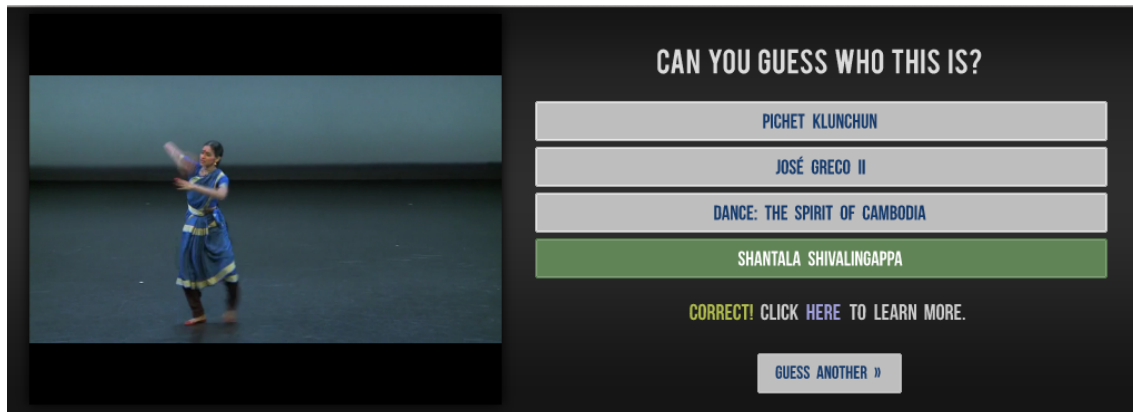


Figure 2.9: Jacob’s Pillow Dance Interactive: Quiz

By analysing this archive, it becomes clear that its creators had an educational purpose when building it. The way in which all the content is stored and displayed, it is done so the person browsing can have a perception of the history of Jacob’s Pillow, the evolution of different dance genres and some of its participants. Using a game to provide a browsing interface through the contents of the archive is truly a creative method, making it interesting to discover works from new or formerly known artists while playing.

Considering the scope of the content and the quantity of items in the archive, the approach used for browsing and display these items is undoubtedly good. It provides a clean and easy to use platform, keeping focus on the context of the results and maintaining its organization. However, in an archive with considerably more content as well as contributors, it could become hard to maintain these two aspects. The lack of a search interface is noticeable as well and, even if in this archive it can be considered a minor fault or even not necessary, in the scope of a work with considerably larger dimensions, it would be indispensable.

2.2 Relational navigation of information

Navigating through great quantities of information is a process which normally requires the user to go back and forth between different categories, sub-menus and listings of the available content [21]. With this in mind, one crucial aspect to have into consideration when designing systems with a considerable amount of browsable information, is to determine **if there is an inherent relation between each element**. There are several situations where this happens and, in these cases, the data elements can be represented

by the nodes of a graph, having the edges representing the relations [23]. Graph visualization is a typical solution for representation of structured or hierarchical contents. In every computer we can find a file explorer which uses file hierarchy in order to represent a tree (which is a special type of graph). In the context of the Web, most of the browsers fail to provide a contextual overview necessary for global orientation. The approach used instead is to provide sets of URL links which connect one item to another. Lai et al. [26] propose a system to generate a graph representing the website map, where the nodes represent URLs and the edges of the graph indicate that two URLs are directly connected (one page contains a link to the other). As the main navigation functionality to be implemented further in this work is going to be based on the relations between the contents, it is relevant to provide a brief explanation on some of the most used graph layout algorithms and graph visualization and navigation techniques.

When the amount of contents to be displayed at the same time is too large, it becomes hard to maintain a clear perception of each individual item and how they are connected. Regarding graphs, the problem consists in calculating positions for each node so that maximum visibility of every node and their relations is achieved. In order to do so, graph layout drawing algorithms are used and, considering the different scenarios of each case, a different algorithm might be better suitable. These can be categorized according to the type of layout they generate. For example, **grid layout** algorithms position nodes in points with integer values, these algorithms consider graph theory concepts such as graph planarity and the goal is to create graphs on the plane with no edge crossings [23]. **Force-directed** algorithms are widely used when representing graphs with large quantities of nodes, as they calculate a position for each node where they do not overlap others [23]. It may be required to further adjust the graph layout in order to display all its content within the viewer's display. It is possible to reduce the size of the graph by using techniques such as **clustering** [23] in order to identify clusters on the sub-graphs and adjust their visibility according to the viewing context (zoom level on the graph for example). Regardless of the type of algorithm used, it is very important that it is efficient and that the resulting graph is predictable so that several iterations of the algorithm generate similar graphs [23, 30]. When implementing a system with graph-based navigation, where the user has to interact with the graph in real-time, both these factors are critical. Efficiency for obvious reasons and predictability so that the user can maintain a mental map of the navigation structure and not becoming disoriented every time they use the system.

It is not so common to see such type of techniques applied either for the navigation of the archive or the media content. The platforms analysed in the following sections are some of the most interesting archives [9, 3] and projects [1] which handle the visualization and navigation of its contents through proximity or relation-based techniques.

2.2.1 Digital Vaults

Digital Vaults is a project associated to the National Archives ¹³ of the United States of America. This fully interactive archive provides access to a great amount of digital versions of documents and materials created during the history of the United States federal government. Among photographs, posters, maps, drawings and others, all of the contents in this archive are identified by tag words which will eventually group them. These tag words are the base for all the relational navigation of the archive. Entering the archive consists in choosing one of the randomly displayed items to be the central object. Upon the choice of the central object this will be displayed, surrounded by all the related items according to the common tags. Figure 2.10 illustrates a situation where an illustration of a Founding Father was selected as the central item, surrounded by a cloud of related content which contain at least one tag word of the list in (1). As the number of items may become very large, depending on the common tags, it is possible to hide those we do not want visible. By selecting tags on the list, the contents with that tag word will be visible and also, by hovering the mouse on one of the active tags, it is possible to see the direct relations between the items. In figure 2.10 the links are showing all the items with the tag "Constitution".



Figure 2.10: Digital Vaults: Related items

Further filtering is also available in the "Filters" tab, providing the possibility to choose the media type, timeframe or relevance of the visible items. Selecting any of the surrounding objects will switch the central item, and all the correspondent related content will be re-arranged. In order to see the details of an item there is an icon (2) which, when used, opens an overlay with the information. This option is also visible by hovering the mouse on any of the related items. There is also an alternative method of exploring the contents which consists of a series of challenges, named *Pathway Challenge*. This option provides an interesting and interactive way of navigating through different items searching for clues in order to complete the challenge. Similarly to the approach used by

¹³<http://www.archives.gov/>

the archive reviewed in section 2.1.2.3, this one tries to provide an educational but still captivating browsing experience.

2.2.2 Dancers Project

Dancers¹⁴ is a project which has the main goal to focus only on the dancers, giving the possibility to freely express their art. It is built as an interactive video database of professional dancers improvising all within a specific context. Besides being an archive for solo performances, which can be browsed like any other, it is also a library of movements where the characteristics of each dancer's performance are analysed and catalogued. This analysis is based on the dancers' movement over time, which is carefully analysed according to a set of previously determined descriptor quantifiers. Characteristics such as movement speed, proximity to the camera, spatial use, among others, are quantified by filters which receive information from two different cameras. This is the basis for comparison between two dancers and consequently for spatial grouping according to the similarities. The performance is thoroughly catalogued according to these factors and 4 characteristics are chosen dynamically in order to distribute the contents spatially as in figure 2.11. This way it is easy to identify and compare performances where they may be similar and thus close to each other by a certain characteristic, but not so identical in another placing them with some distance between them.

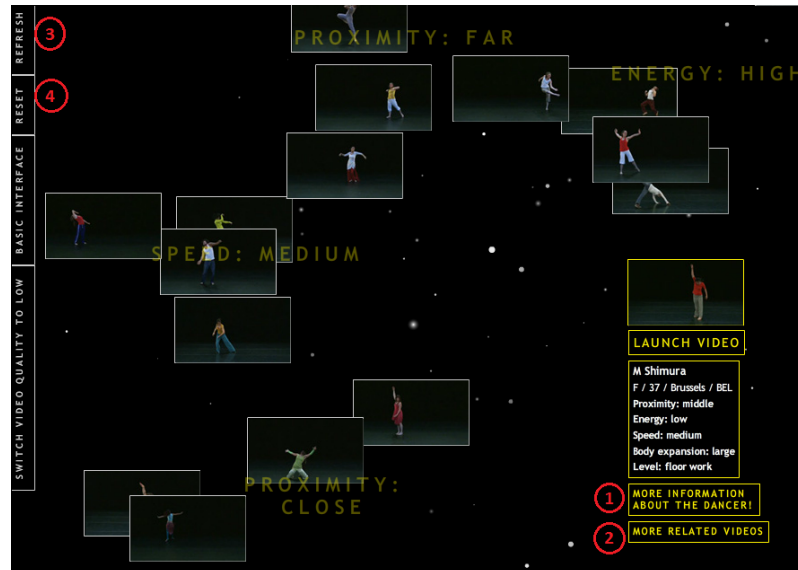


Figure 2.11: Dancers: Relational Navigator

The video of the performance is easily accessed and it is visualized in an overlay over the current display. This is useful when the user intends to further compare dancers or performances, as it is not necessary to go back and forth between each dancer's personal page. In order to obtain more detailed information about a dancer and/or a performance,

¹⁴<http://www.dancersproject.com/>

the user may click in (1) and a new window will open for the specified content. It is also possible to place an item in the center (2) in order to compare others to it, also according to random characteristics. In order to obtain different combinations of comparative characteristics the user can refresh the display by clicking (3). This will shuffle and distribute the contents according to a new, random set of characteristics. Finally, clicking in (4) will remove the central item (if there is any) from focus and return to the normal comparative display, as well with a random set for comparison. In the dancers' detailed visualization it is also possible to find related performances, not in the same spatial distribution, but selected according to their common characteristics.

In addition to the relational navigator, there is also the typical browsing and filtering capability, where it is possible to search for a specific dancer, filter by gender, country, dance style, among others. Dancers *Relational Navigator* is an interesting approach to the concept of navigation between related contents. There are, however, a few details to note. Firstly, it would be good to have the possibility to display all the available content considering all the descriptive characteristics, creating thus a global comparative overview. It is also a weak point on the archive the fact that it is not possible to choose by which factors the contents will be filtered. Finally, it would be a great improvement if the results generated were retrieved in considerably less time. As mentioned in 2.2, it is important that the navigation of results in this kind of relational layouts is done as close to real-time as possible. The amount of waiting time for the results will have great impact in the overall usability of the archive. However, it is important to note, that this type of comparison is based on very specific data which involve complex calculations in order to retrieve the results. Applying this navigation method supporting another type of data, such as the typical tag words, might result in an interesting and effective solution.

2.2.3 Bestiario: Videoshpere and Bestiary

Bestiario is a company that focuses on the development of applications for the analysis and management of complex relational information, focusing on dynamic data representation and the creation of interactive spaces for the collective creation of knowledge. By combining creative designs with concepts of graph theory, geometrical and geographical representations among others, they are able develop some interesting and natural applications for multimedia information representation and navigation.

The examples that are detailed here are those which prove to be the most relevant in the scope of this work, and could provide some implementation ideas for the system to be developed. The first one represents multimedia data with visible relations through common characteristics and the latter is a good solution for the visualization of graphs with large quantities of information and relations on a limited viewing area.

The Videosphere ¹⁵ is a collection of TED Talks ¹⁶ gathered to form a large multimedia knowledge network. The contents of this application are solely in video format and are related using semantic metrics and compatibility. In order to measure a relation between two videos the there are some calculations involving the number of common and distinct tags between each video, which will define their proximity.

The amount of relations between the videos may cause some visibility problems, as in figure 2.12 so, in order to avoid this, it is possible to adjust some aspects of the sphere's display. The controls in (1) enable the users to increase or reduce the size of the sphere, toggle the visibility of the relations between the videos and change the visualization mode from outside to the inside of the sphere. A set of tag words is also provided (2) in order to facilitate the search of specific types of videos, whose results will appear right bellow it. The navigation within the sphere is done by clicking on the desired video or on the arrows which indicate the path to a related video.

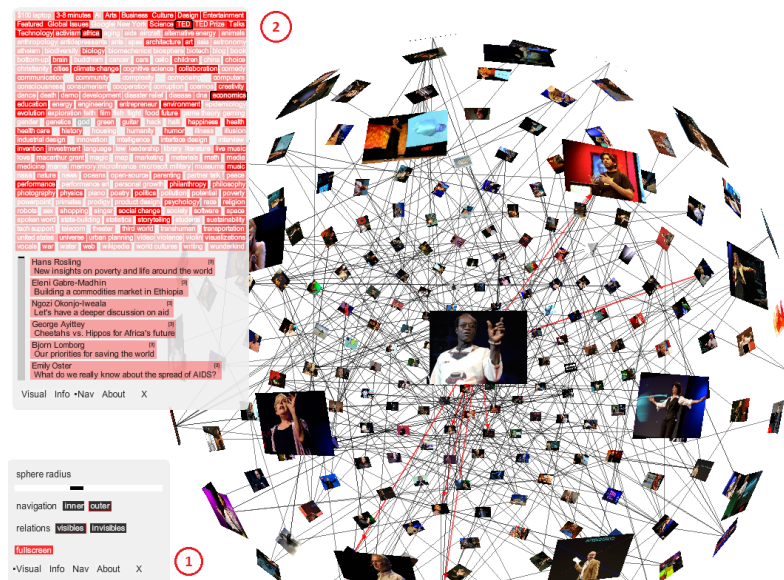


Figure 2.12: Videosphere: outside view

As mentioned earlier, the amount of videos and relations between them may cause visibility and navigation issues. Even though the view from outside the sphere is useful for randomly browsing all the available content, it can become really hard to perceive the existing relations between the items. In this situation using the sphere's inner view proves to be more efficient. Figure 2.13 illustrates the inner view of the sphere, with the same video selected as previously, but with the relations linking to other videos fully perceivable (1). In order to explore content related to the current video it is only necessary to click on one of the arrows, and the sphere automatically shifts to the video correspondent to the relation. The box in (2) may display the information of the currently selected

¹⁵<http://www.bestiario.org/research/videosphere/>

¹⁶<http://www.ted.com/>

video, including the link to the source of the video, the list of tags which describe it and a small description. Additionally, either when hovering on one of the arrows or on top of a nearby video, a description of that video will also be displayed in (2), together with and indication of the common tag words between them. The visualization of the video content is done in the same way on both view modes, by simply clicking once to expand the video and again to start the playback.

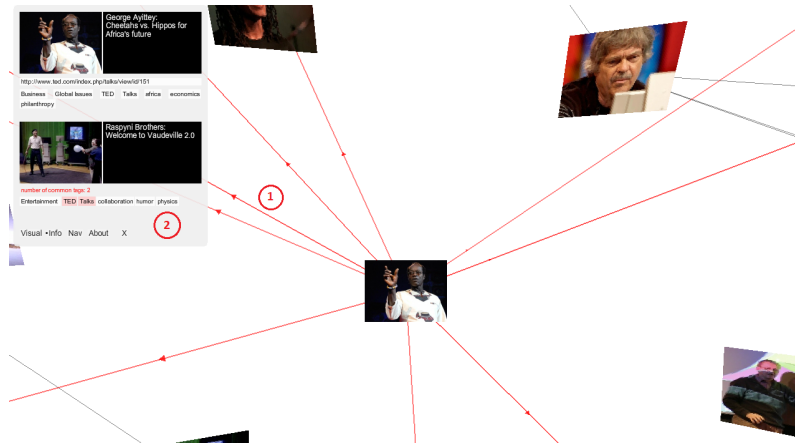


Figure 2.13: Videosphere: inside view

The Videosphere serves as a good example on how the existence of visible relations between media content can improve browsing experience. However, it raises some issues considering the visibility and navigation usability when the dimension of the archive becomes considerably large. In order to solve these issues, some graph navigation techniques may be used, as briefly explained in the beginning of section 2.2. This is what happens in the second example analysed. Bestiario Bestiary¹⁷ is a simple social network representation of people related to the company's origins. It consists of a graph layout with nodes representing both the person and related area or department. The links forming the relations are only between each person and the related areas, and the position of the person node shifts in the direction of each area, depending on the weight of the relation. All the nodes are movable and the people's nodes open a small window with information about the corresponding person when clicked on. Figure 2.14 shows the network obtained, in its initial layout, with a few information windows open and the links pointing to the source node.

¹⁷http://www.bestiario.org/_proyectos/past/

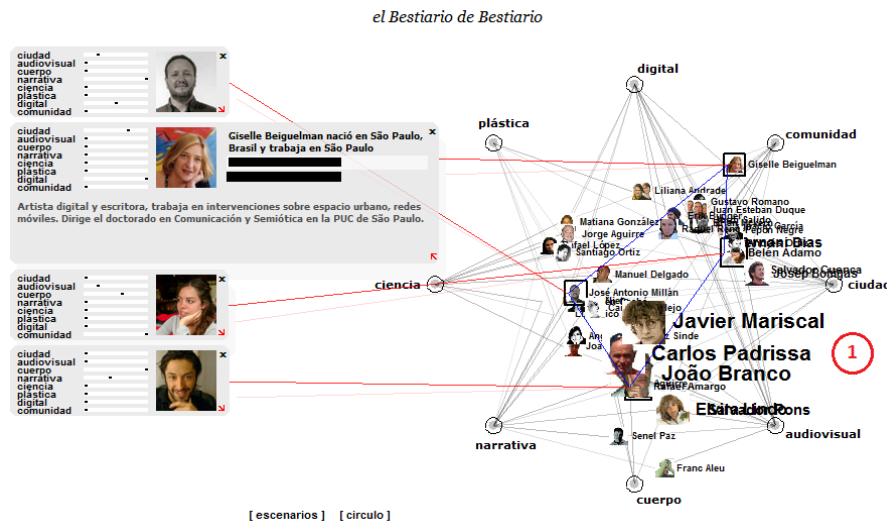


Figure 2.14: Bestiario Bestiary

As mentioned earlier, this network's graph is a good example of how the visualization of the nodes may become difficult when the number of items and relations is large. The solution found in this case is the use of the Fisheye Distortion technique [23]. What this technique does is zooming in the graph's visualization on the location where the mouse is located (1), resulting in a clear view of the nearby nodes. In the scope of this thesis and the relational navigation system to be developed, this solution together with the use of clustered sub-graphs [23], might be a suitable solution for implementation.

2.3 Timelines and video annotation visualization

Video is a rapidly growing media format throughout the Web. With the increasing amount of available tools for the creation and sharing of digital video content, it becomes accessible for any person to produce personal videos. Thus, in order to improve the management of video archives and content retrieval, it is necessary to describe and categorize such information. This commonly results in the creation of metadata or annotations to support the content description. As such, the study of video annotation has come to a point where there is a need for a common metadata format. This led to the creation of the MPEG-7 standard [8], which is used by many video annotation platforms. This standard defines an alternative set of descriptors for audiovisual content, enhancing the indexing and search of this type of content.

There are already many video-based systems which implement text annotations. In YouTube¹⁸ users have the possibility to annotate their uploaded videos, which will be visible for everyone else. In this case the annotations will appear on the desired portion of the screen but there is no indication about the time when they happen in the video

¹⁸<http://www.youtube.com/>

timeline. Viddler¹⁹ on the other hand, enables users to post comments anywhere during the video and these will be indicated as a tag in the video timeline. Providing a global overview of the locations of annotations within the video is important to provide an efficient video browsing experience. However, this may lead to issues regarding effective display of information, both on the timeline and over the video, or effective user navigation throughout the annotations as referred by Costa et al. [20]. Muller et al. [27] emphasizes as well the importance of effective filtering mechanisms to support annotation visualization. This relates to how the user can control the visible information, both on the timeline and the video, in order to maintain a cleaner display of contents.

With the implementation of multiple layers of annotations in which two annotations may appear at the same time, some visualization issues arise. Beginning with the types of annotations supported, as well as for the global overview of the annotations within the video timeline and how the annotations itself are displayed, there are numerous projects [20, 29, 22, 25] which attempt to provide effective solutions. However, in order to further analyse any video annotation system, it is important to consider both the scope of this work as well as the content types to be supported by the archive. Therefore, the following sections will detail how the systems which will export content for the archive approach these issues, and analyse one particularly complex system with support for video annotation.

2.3.1 ELAN

ELAN²⁰ is a linguistic annotation tool which provides the possibility to create and browse through textual annotations on both video and audio sources. Although the tool was designed specifically to support linguistics analysis, it can as well be used for any other area. The system enables the association of, at most, four videos to the same annotation file at the same time. With simultaneous visualization of the media, it is possible to synchronize their playback individually, making it a useful system for multi-camera recordings. It is important to note the relevance of this system within the scope of this work. Being a platform used by some of the future users of the archive to be created, it generates one of the main content types to be integrated. Considering the specific implementation details of annotations in ELAN, which will be analysed further on, it is essential to consider how these are displayed both for the annotations themselves, as well as within the context of the video timeline.

Annotations can be created in multiple layers, referred in ELAN as "tiers". These tiers can be designed hierarchically and each annotation can be either aligned to a time frame or refer to another annotation. The possibility to create multiple annotation layers, in which

¹⁹<http://www.viddler.com/>

²⁰<http://www.lat-mpi.eu/tools/elan/>

various annotations may be assigned to the same time slot, results in a complex visualization layout when compared to other systems. Figure 2.15 represents an example of the overall interface layout. There are two visible video timelines. Both of them enable the navigation through the video although in a distinct approach. The first is a simple timeline (1) with a cursor which indicates the current position within the video. It has a fixed size displaying the complete length of the video, which enables a fast navigation to any point within it. The other timeline (2) displays all the created tiers, as well as an overview of the annotations contained in each one. The purpose of this timeline is to provide an overview of the existing annotations and its duration within the video, providing as well an easy way to navigate through them individually.

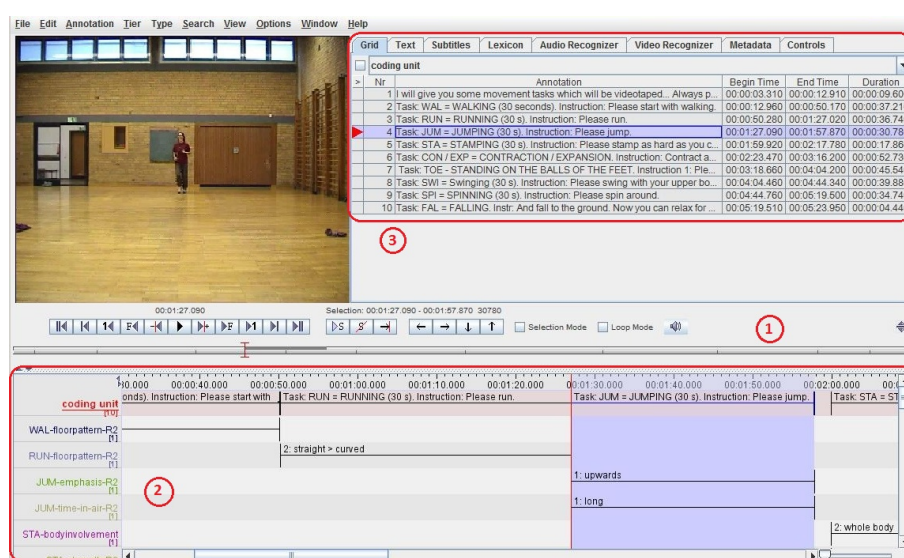


Figure 2.15: ELAN: Overall interface

The mapping of all the annotations and corresponding tiers is saved in a specific XML file created by ELAN (*.eaf, "EUDICO Annotation Format"). Although this is the predefined format for the annotation file, the system is also compatible with types of files generated from different annotation software [6]. Having support to XML format, together with the extended exporting flexibility, makes it a simpler task to create systems supporting this type of content.

As mentioned earlier, the amount of tiers and their respective annotations and the fact that there could exist annotations from multiple tiers at the same time, requires a different approach to the way they are displayed. In order to preserve the distinction between tiers there is the normal display of annotations (3), where the user needs to select the intended tier and the corresponding annotations will be displayed. There is an automatic update indicating when an annotation appears at the current time, as well as the possibility to click on one of them in order to jump the video to the correspondent time. As an alternative to the individual visualization mode, there is as well the option to view up to

4 tiers simultaneously. By selecting the tab "Subtitles" in (3) the annotation visualization will change to the layout illustrated by figure 2.16.

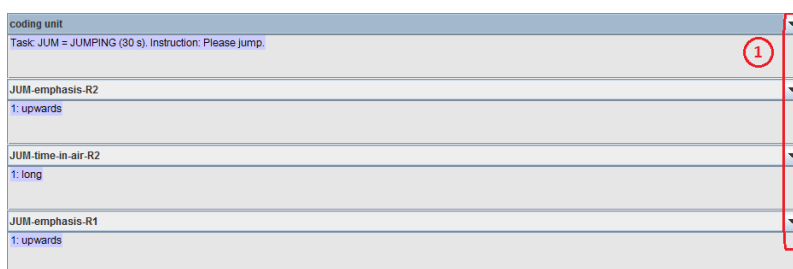


Figure 2.16: ELAN: Multiple tier view

This alternative is useful, as the name implies, when working with subtitles in different languages. Having multiple tiers displayed at the same time provides an efficient way of comparing their contents, and eventually edit them. In order to select which tiers to display, the user simply has to choose them individually from the drop-down lists (1), in the desired order.

In order to increase the support for different media formats as well as high quality videos, without sacrificing the system's performance, ELAN relies on external media frameworks such as DirectX and QuickTime [33]. This implies that the possibility to use certain video formats will depend on whether the underlying media framework supports it or not. It is worth noting that exists a web-based version of the software called ANNEX [28]. This version of the tool is implemented in Flash and makes use of the browser's multimedia plugins in order to play the videos.

Considering the complexity that can result from multiple annotations being displayed simultaneously, it is necessary to carefully analyse how the contents exported from ELAN should be presented within the archive. Having too much information, as happens with ELAN, could be counterproductive. On the other hand, not displaying enough information would result in loss of context of the annotations within the video.

2.3.2 Creation Tool

The Creation-Tool [19] is a video annotation tool which was created within the scope of the TKB project. With the aim of supporting the creative process in the performing arts, the Creation-Tool provides real time video annotation support which can be used, for example during rehearsals of contemporary dance pieces. The tool was designed for Tablet PCs, with support for touch digital pen interaction. It provides support for various types of annotations, which can be divided in two distinct categories: **graphical** and **audio** annotations. The graphical annotations include: digital ink, text, hyperlinks and marks. These annotations can be placed within or outside the video's display area and

can be fully customizable by the user. Audio annotations simply consist of external voice recordings.

It is possible to annotate the video in three distinct modes: continuous, delayed and suspended. The first two work in a similar manner where the user annotates while the video is running, focusing on short and quick annotations which will be shown as segments within the timeline. The only difference is that the delayed mode plays the video with delay, giving the possibility to first observe the live action and further annotate when the video plays the desired scene. Suspended mode focuses on more complex annotations within a specific moment and only refer to a single frame in the video.

In order to control video playback and navigation, the Creation-Tool has two different timelines (Figure 2.17). The video timeline 2.17(1) serves simply for video navigation, together with the video player's controls as we. The other one is divided into two different tracks. The first consists of a track of thirteen video frames 2.17(2), updated along the video playback. The latter is the annotations timeline 2.17(3). It provides an overall view of the annotations created within the video, separated by the different annotation types. Figure 2.17 illustrates a situation where all the graphical annotations (mark, ink, text and link) are visible at the same time.

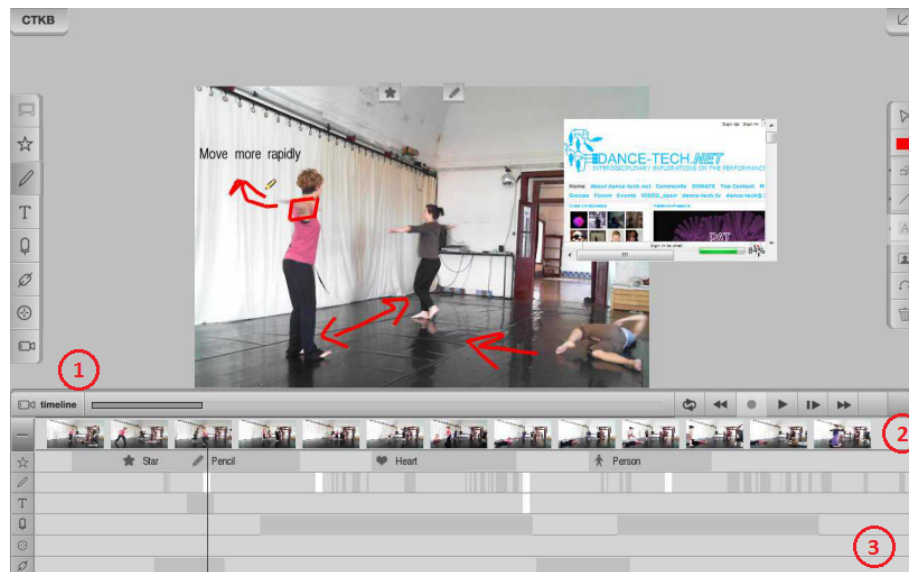


Figure 2.17: Creation Tool

The Creation-Tool was implemented in C++, with resource to the following platforms: OpenCV ²¹, OpenFrameworks ²² and Qt Gui ²³. Likewise, the multimedia handling is managed with help from these platforms, resulting in a video file in Quicktime format (*.mov) and separate annotations file in XML format, with the corresponding mapping

²¹<http://opencv.willowgarage.com/wiki/>

²²<http://www.openframeworks.cc/>

²³<http://qt.nokia.com/>

of the video and annotations. It is important to emphasize that the archive will support the visualization of contents exported by CT. Therefore it is good to note that the fact that the resulting video is separated from the annotations enables the re-usage of this content in a more flexible manner.

2.3.3 Synchronous Objects

Synchronous Objects²⁴ is an interactive web project of the choreographer William Forsythe and the Ohio State University. It focuses on the reproduction of Forsythe's complex ensemble dance "One Flat Thing", into an unique collection of on-screen visualizations. The project aims to examine how the ideas in a choreography can be transformed into visual objects which will enable an easy understanding of such concepts, as well as suggest new interpretations of the dance. The dimension and scope of this project's research is quite large and contains concepts which would be both to extensive and irrelevant to detail, considering the scope of this work. As such, the following analysis will focus only on how the annotations are displayed, both on the timeline and the video display.

As mentioned earlier, the quantity of details in the analysis of this choreography is very large and the resulting timeline layout shows just that. Therefore, in order to provide a simpler analysis of the timeline, some of the object were disabled from visualization. In figure 2.18 it is possible to observe the video overlay annotations, which follow the movements of each dancer throughout the video, as well as the timeline annotations separated in different layers for each dancer. The timeline in this system is subdivided into

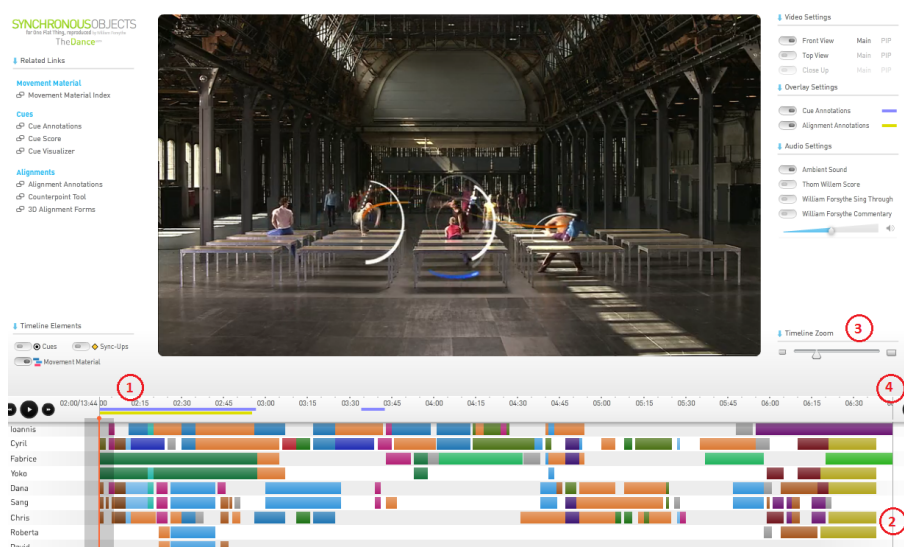


Figure 2.18: Synchronous Objects

two distinct but connected parts. The first (1) is used to display the overall length of the video overlay annotations, regardless of the person. In the other part (2) there are several

²⁴<http://synchronousobjects.osu.edu/content.html>

distinct tracks, one for each dancer in the choreography, each with their specific movement related annotations. It is possible to adjust the length (3) of the visible timeline by zooming, in order to do a detailed analysis or get a more general view. By hiding the bottom part (4) of the annotations timeline the video expands and all the filtering controls are hidden, thus providing a better focus on the video itself and the overlay annotations.

Regarding a more technological analysis of the system, it is important to note that the visualization is not smooth, being noticeable constant delays during the video playback and seeking forward/backwards. This may relate to the fact that the system is implemented in Flash, which is known to generate resource-heavy web platforms. This allied with the amount of dynamic content to be displayed proves to be a poor candidate for the intended implementation purposes of this work.

2.4 Summary

In this chapter we presented an overview of the most popular multimedia web archives, as well as few dance archives relevant in the scope of this work. Next, a brief look on relational navigation concepts in general and how it can be applied to multimedia archives, followed by a review of some of the related work. Finally, we reviewed video annotation display and respective timeline representation by analysing some of the platforms used for this purpose.



Archive and relational navigation

Online multimedia archives are one of the most popular ways of storing and sharing content on the Web. With privacy and visibility settings, together with the capability of categorizing each piece of information, these multimedia archives are able to create networks of related content, which can be used to enhance the browsing experience. With this concept at its foundation, the purpose of this work was to develop a multimedia archive with focus not only on the display of the content itself, but as well on the relations between all the visible information.

In contrast to the common search and browsing methods, the archive's tools provided for these tasks should give the users the possibility to research any category or theme existent within it, regardless of the type of media available. Further in section 3.2 will be detailed how such archive is to be organized and the multimedia types supported within the context of this work, followed by an overview of the relational concepts and tools used for its navigation in section 3.3.

3.1 Project context

In order to further explain how the archive is structured and its functionalities, it is important to revisit the context of the project for which it was developed. As referenced in section 1.2, this archive was developed in the scope of the project TKB - A Transmedia Knowledge Base for contemporary dance and its main focus is to provide the tools necessary for any artist to store and display his work online. As it is common for different people to have distinct preferences on how their contents should be displayed, one of the main concerns was providing the possibility of customizing the way it will be stored and

consequently displayed to the users.

Even though the archive is designed and implemented to be generic and thus giving the possibility to be used in any other area or context, there were some important aspects regarding the types of media to be supported and how they are to be displayed. If we consider again the scope of this project and *Creation Tool*, detailed in section 2.8, which is to become one of the main sources of content of the archive, and the amount of videos already annotated by collaborators using other platforms such as *ELAN*, analysed in section 2.3.1, it is easy to realize that some custom visualization options would be necessary for these, regardless of how useful they could be in any other context.

Finally it is good to note that people involved in contemporary arts tend to have distinct definitions and concepts, which are not always easy, both for the audience or user to understand and for other creators to relate to their own art. This is where the relational visualization and navigation of the contents implemented in the archive has its core, transforming these unperceivable connections and shedding some light on how users and contents are related to one another.

3.2 Archive structure and contents

The concept of Knowledge Base was always present in the development of the archive. Stored contents are always connected in some way, creating the aforementioned network which is the foundation of any system with researching purposes. Even though these contents may have distinct sources and been uploaded by different users, which is reflected in the individual structure of each archive, it is possible to create a layer making it invisible to the user.

As this is a multimedia archive there is support for storage and display of video, image and sound, as well as the annotated video types mentioned earlier. Although, even if the system enables a certain amount of customization of each individual archive's structure, it was important to maintain consistency in the design of the overall archive. Thus, as will be detailed in section 3.2.1, some predefined templates or content types were created supporting the different multimedia types which still allow room for customization while maintaining a fixed design.

3.2.1 Content types

Templates, or content types, were created in order to provide a variety of multimedia contents and customization options to each one, whilst still maintaining a certain amount of visual stability throughout the whole system. Therefore the first step before adding any content, is to choose one of the following options:

- **Document Template:** typical text editor, with the possibility of including text, images and video within the same area;
- **Video Player Template:** video player with a simple description;
- **Slideshow Template:** image slideshow of uploaded files;
- **Gallery Template:** image and video gallery with overview and slideshow mode;
- **CTKB Template:** contents exported from the Creation Tool;
- **ELAN Template:** contents from ELAN annotation software.

Considering that the document, video and slideshow templates are considerably simple and very similar to what we can find in other systems, there is no necessity of further detailing its functionalities. Therefore, only the other three templates will be further analyzed.

3.2.1.1 Gallery template

The gallery template implements a design similar to most online multimedia galleries, displaying a global overview of the contents, represented by its thumbnails which when selected will open an overlay with the original content, in a slideshow mode, as illustrated in figure 3.1. This mode is the only option for the individual visualization of each content within the gallery, supporting images, video files uploaded to the system, or videos from web sources such as YouTube or Vimeo.

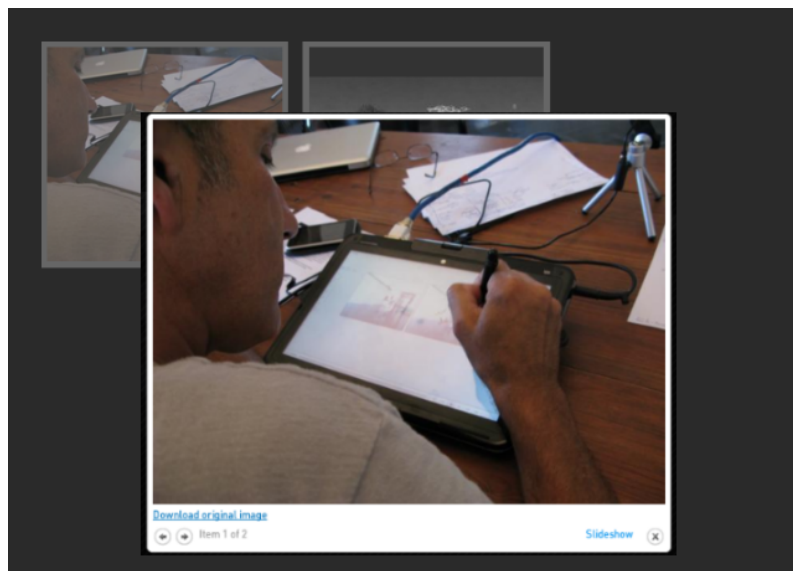


Figure 3.1: Gallery Template

As it happens throughout the rest of the system, the owner of each content has special permissions and therefore options which are not available to the other users. In this case

it is possible to: alter the order of the gallery's files while in the global overview mode; edit the information of each item individually, including title and tags; edit the global information of the gallery as any other content type, including title and description.

3.2.1.2 CTKB template

It has been mentioned earlier that the Creation Tool contents would be one of the main focus of the archive and as such, the template created in order to display them is the most complex and unique in the system. Starting by revisiting the functionalities of the annotation tool, it is good to emphasize the types of annotations it provides, which are: **Ink**, **Audio**, **Text**, **Mark** and **Link**. These types of annotation are all supported by the archive and are identified by the colored squares as shown in figure 3.2. Note that these colors are customizable by the content owner, while creating or editing it. Refer to section 3.2.2 for details on the content creation and this particular aspect of customization.

In the Creation Tool the annotations are originally displayed in the exact position they were drawn, which does not happen in the archive. In order to maintain the focus on the playback of the video while showing the annotations, it was decided that the only ones to be shown on the original position would be marks and digital ink. Everything else, including marks which are annotated outside the video area, are displayed on the appropriate area on the left sidebar.

The implemented timeline was designed to be as similar to the one in the *Creation Tool* as possible, with the goal of providing an overview of all the existing annotations throughout the video. It can be seen as five separate timelines, one for each annotation type, and with the matching color as the sidebar in order ease the association between the time frame and the annotation content, as well as distinguish the annotation types between them. The length of the annotation within the timeline correspond to how long it lasts in the video. While the video plays, the annotations corresponding to the current time will be displayed, as well as highlighted in the timeline. Due to the limited space for the timeline, it was not possible to keep all the annotations visible at the same time, whilst still maintaining a size for the annotation squares allowing the clear distinction of the different annotations and their lengths. Therefore, the solution found was to define a minimum size and, when the current time reaches the middle of the timeline, this starts to scroll in order to show the remaining annotations.

Considering that the amount of annotations in one video can be significantly large, or the fact that different users might have no interest in a specific annotation type, lead to the implementation of a filter within the playback. This filter, visible on figure 3.2, uses once more the color scheme for the different annotation types, and can be used to show

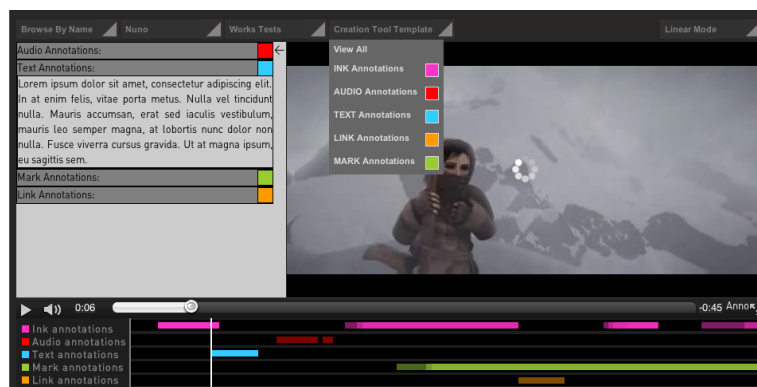


Figure 3.2: Creation Tool Template

and hide the chosen options. In addition to this and the normal fullscreen mode, it is also possible to hide both the sidebar and the timeline, in order to give more focus to the video, having automatic adjustment of both the video and the annotations to the current display size.

3.2.1.3 ELAN template

This template is similar to the former in regards to the way the annotations are displayed on both the sidebar and the timeline. Given this, the main differences between the two templates are the annotation types. While in the CTKB we have five different types of annotations with distinct visual characteristics, in ELAN we only have one type, which is **text annotation**. The singularity about ELAN annotations is that they are separated into predefined categories, once again identified by the color scheme, which can themselves be divided into sub-categories. As the number of sub-categories may be substantially large, displaying all of them at the same time would result in a confusing design, especially regarding the timeline. The solution found is illustrated in figure 3.3.

The annotations start by appearing all grouped into the main categories both on the timeline and the sidebar but, if desired, it is possible to view the sub-categories of a specific main one, with resource to a filter similar to the Creation Tool template. The normal functionality is maintained, showing or hiding the selected options although, when only one of the main categories is selected, it automatically expands to show all its sub categories, separated by the proper names. Likewise, there is the possibility to completely hide the sidebar and the timeline as well as change to fullscreen mode, in order to focus solely on the video.

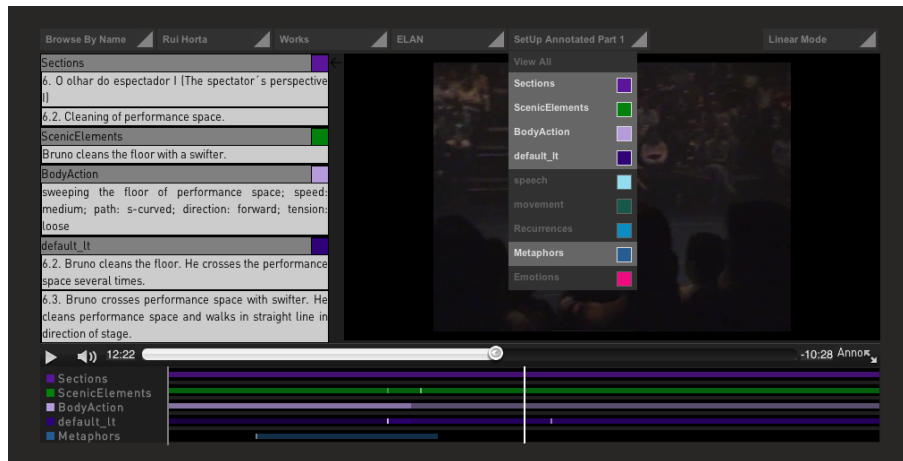


Figure 3.3: Elan Template

3.2.2 Structure and organization

The global archive can be described as a set of smaller individual archives created by users, which will all be interconnected by the concepts, tag-words or categories defined by its creator. The structure of the individual archives is a simple tree hierarchy based on the menu entries of each content and it is limited to a certain depth, in order to maintain a clean and consistent design throughout the main archive. It is important to note that each archive's root level is automatically created, corresponding to the name inserted by the user when creating the account, being possible to change it if desired.

3.2.2.1 Creating content

When logged in, a user has access to the administration part of his archive where it is possible to either customize the existing structure or create new contents and extend the archive. In order to do this there are predefined structures that users can add to the archive, which we can categorize as either **content** or **menu** structure. The former consists of the actual contents of the archive and, as defined in 3.2.1, may take a range of content types. The latter, as the name implies, refers to the intermediate menu structures which will serve both as containers for the actual contents, and as a mechanism to maintain the structure of the individual archives as identical as possible.

The actual process of creating new items, either menu or content, differs in the number of steps required. In order to add a new content item, it is necessary to first select which content type is intended. Menu items, however, do not have a content type definition and as such it is not required to go through this step in order to create them. Following this, the user proceeds to upload the actual information for the specific item, which is divided in the three following steps:

1. **Menu Settings**
2. **Meta Description**
3. **Add Content**

Even though each content type, or template, is determined by the different fields it requires, all of them along with the menu items share two very important structural fields, the menu entry information and the meta data, which correspond to the first two steps. The first one is quite self-explanatory as it regards the name of the menu entry, is the name which will be displayed in the archive structure for this particular content, visible by every user. Moreover, if this content had been created earlier and opened later in editing mode, there is also the possibility of altering the parent menu item on the tree structure, as long as the restrictions mentioned earlier are respected. The second step is the most important regarding the relational properties of the archive. It is here that each content can be tagged or categorized with either already available expressions, used in other contents or by other users, or by new words which will be added to the existing list.

As mentioned before, there is an imposed depth limit in the archive's tree structure. Whenever one of the menu items is selected, the respective available options are highlighted in the bottom menu buttons, resulting in a maximum of three levels where contents are always placed at the bottom. Figure 3.4(a) represents an individual archive, before and after the contents have been added. The restrictions of the available options are also noticeable in the figure, having only the "CONTENTS" highlighted in the bottom menu, given that the next item to be created is at the maximum level permitted.

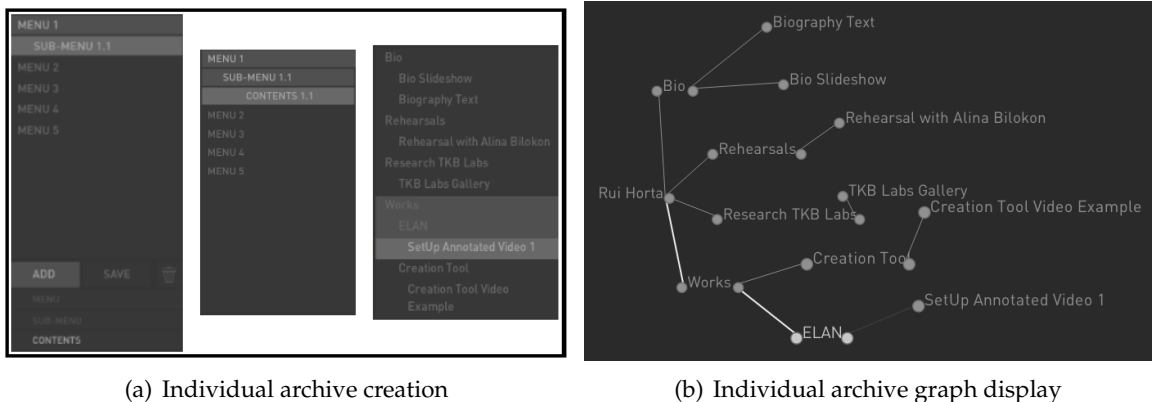


Figure 3.4: D3 hierarchical layouts

As the contents are uploaded to the archive the changes are immediately reflected, both on the individual and global archives. The tree in figure 3.4(b) represents the individual archive of a user when all the levels are visible. This view is the same for both the owner

of the archive and visiting users, reflecting the changes done in the administration mode and thus, providing the owners a preview of how the archive will look after the changes. The goal of this example is solely to illustrate how the contents are organized within the individual archives and how the changes in their structure can easily be reviewed. Section 3.3 provides further explanation of the browsing and searching capabilities of the system.

3.2.2.2 Create content: annotated videos

Both of the annotated video templates supported by the archive share the same process of handling the annotations, regarding individual annotation visibility and the color scheme by which annotation categories are identified. It is important to note that the two main fields these templates require are the original video and the text document containing the annotation details. Although the structure of the annotation document is mostly different, the type of information we intend to store about each annotation: color and visibility, is the same for both templates. Figure 3.5 is an example of ELAN annotations being edited, which are listed according to the categories defined in the document. The options included allow the user to choose which annotations are visible during playback, as well as select a color to identify the respective category in the sidebar and timeline, as detailed in the previous section.

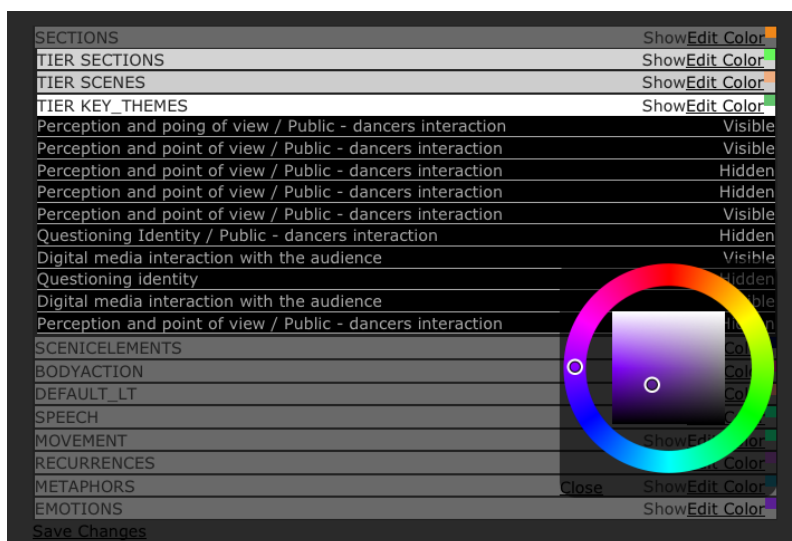


Figure 3.5: Edit annotations: ELAN

The main difference between the two templates is, as detailed previously, how annotations are categorized or grouped, and this reflects on the way the annotations are listed in the editing mode. For the CTKB template, instead of having the nested categories, there is a listing of the annotation types with the correspondent annotations, with the same options as in figure 3.5.

3.3 Relational navigation and search

In the previous sections there were some explanations on how the contents are related to one another and in what way can the users alter this to their own preferences. In this section, however, the goal is to give an overview of the global archive navigation, how the relations between the contents are used to create such connections, and in what way this can be used to develop relational searching functionalities.

3.3.1 Archive navigation

Earlier in section 3.2.2 we defined the global archive as a set of smaller, individual archives related to each other by categories or keywords set by their owners. However, it is important to distinguish the type of navigation available for each one.

The global archive is the landing page for all the users, authenticated or not, representing an overview of all the contents and users, as well as the connections between them. This way, somebody who is unfamiliar with the archive or the contents available, can easily start exploring, jumping directly into any user's personal archive or individual content items. Technically the global archive is a graph, whose nodes initially represent the names of the individual archives, and links representing the tags or keywords which categorize each one. Additionally to the initial graph, users have the possibility to choose from a list of content categories, which will generate a graph having content items which fall in such category as its nodes. Moreover, in order to provide a filtering mechanism for the results displayed by the graph, there is a list of available keywords provenient from the result items, like the one on figure 3.6. Toggling these keywords will instantly show or hide the matching nodes and links. Note that some archives or contents might not have any tags associated to them and consequently will appear as an isolated node with no links to any other.

Hovering through either the links or the nodes displays a list of tags associated to that item, providing a global overview of the context in which contents are related to each other. Moreover, when the user who is browsing is familiar with a specific archive or content, the possibility to know how many common tags between each node by hovering the links between them, provide a simple manner of discovering contents this user might actually be interested in.

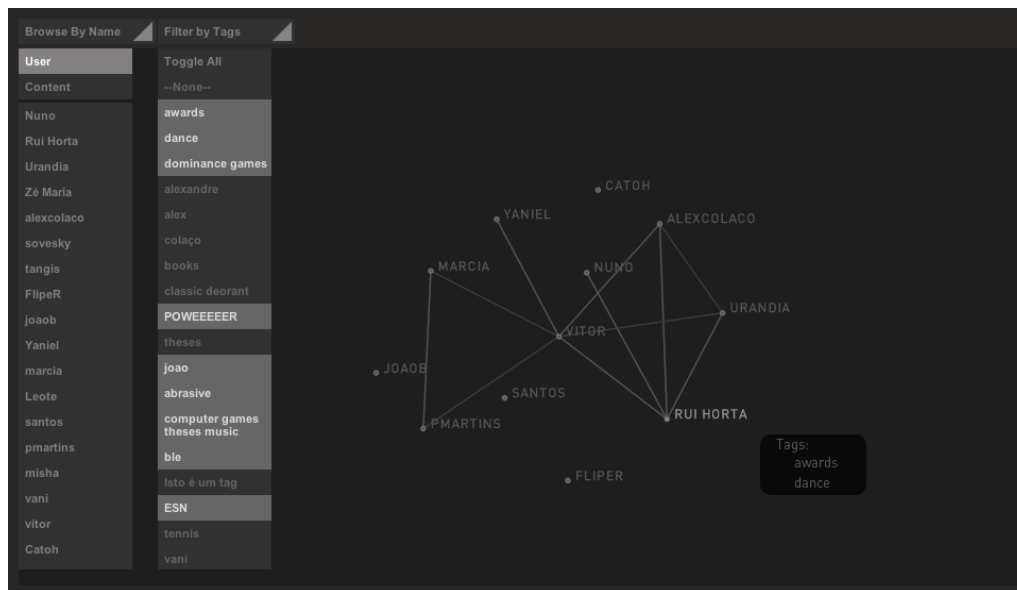


Figure 3.6: Global archive

3.3.2 Individual archives

Accessing the individual archives' main page is the only way of browsing through its contents in the hierarchical manner they were created by the owner, as previously detailed in section 3.2.2. Once again, the individual archives are built as graphs. However, the connections defined between each node result in a tree structure as represented in figure 3.7. Unlike the ones in the global archive, these links only serve to represent the hierarchy in the archive, as well as the active path of navigation. By selecting one of the nodes of the graph it will expand, revealing the underlying nodes, which may be either menu or content items. While selecting nodes to navigate the archive the active path will be highlighted, showing both in the navigation menu 3.7(1) and on the graph itself 3.7(2), represented by the different colored nodes and links.

Following the same concept of the global archive, hovering through nodes brings up an overlaying menu box 3.7(3) which allows either to view the full content of the node, or to search for other contents related to it. In alternative to the graph selection, it is also possible for the user to go to the intended content by selecting the respective menu item 3.7(1). Viewing and navigating the individual archive this way is defined as the "Cloud Mode", which provides the relational navigation functionality. However, when a content item is selected, the system enters into a different display mode, called "Linear Mode". This visualization mode consists essentially in displaying the contents as they were defined in 3.2.1, but with a small particularity regarding the menu options in 3.7(4). These appear in both modes allowing the user to toggle between them at any time, and always regarding the current navigation path. This means that the graph will automatically expand until

the node corresponding to the content which was being viewed in linear mode, contextualizing the user once again in the relational navigation and the cloud mode.

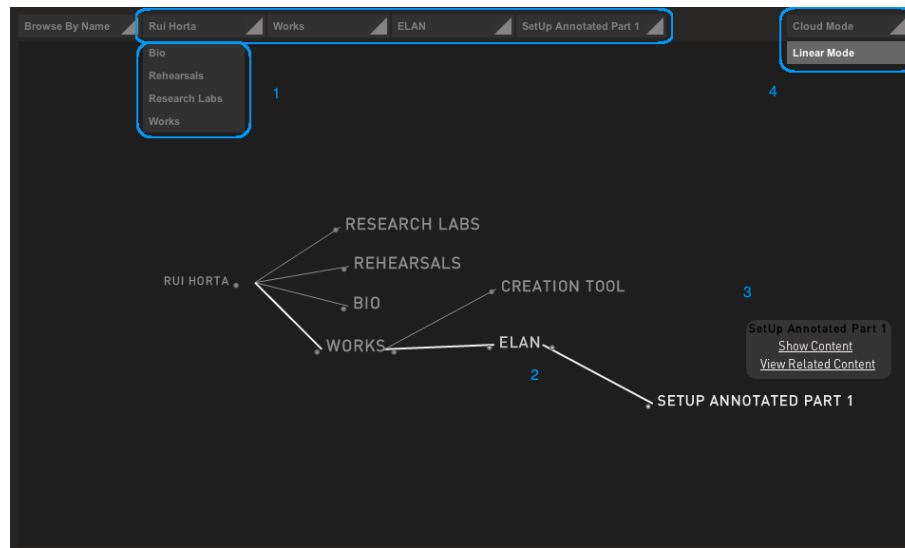


Figure 3.7: Individual archive navigation

Note that the archive was designed with special attention to the concept of relational navigation and thus the "Cloud Mode". What this means is that the linear navigation of the global archive is not defined extensively, being in most occasions reduced to a listing of the content links in each individual archive and its intermediate container pages.

3.3.3 Search and related contents

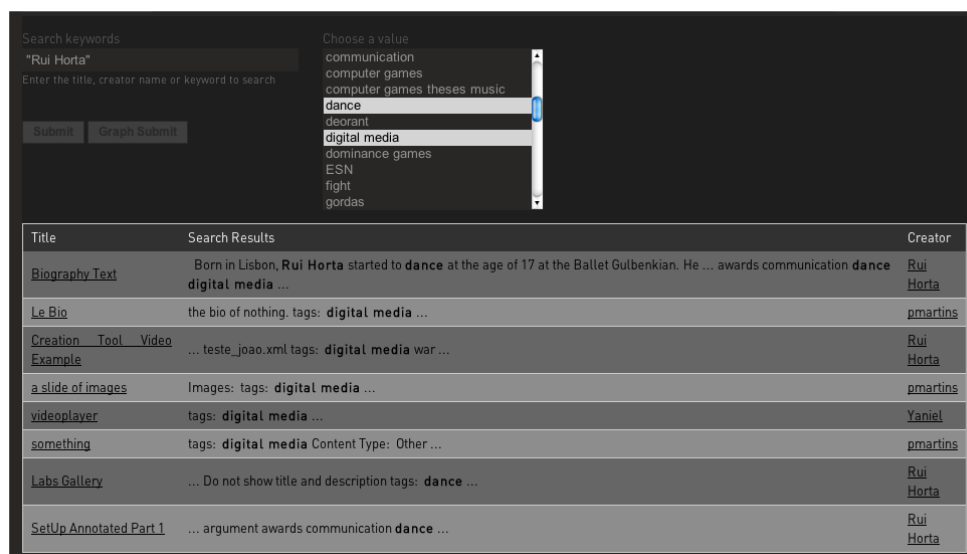
The designed searching mechanism is one of the key elements of the related contents browsing throughout the archive. Although the graph of the main archive displays all the distinct connections between the various contents or users within the system, its filtering options are limited to the ones predefined in the system, and also do not provide a natural way to filter contents which do not relate to a certain node other than selecting the tags associated to it one by one. In order to do this it was decided to gather the concepts of the regular searching methods found in most archives or websites, with the graph and relational content described in this work, resulting in a flexible and unique set of searching and browsing options.

The searching interface provides two options for the searching input, as well as a combination of the two:

- **Keyword search**
- **Tag selection**
- **Tag and Keyword search**

Starting a keyword search gives the possibility to enter any word, relating to the content title, words used in texts as well as tag words. Note that multiple word input is supported, providing the best option for a more specific and detailed search, where the user already has some idea of which contents he is looking for. Tag selection is a mechanism similar to the browsing of the global archive 3.3.1, where the results of the search will be contents which are categorized by the selected tag words. This searching method, as referenced before, is mostly useful for browsing the archive in a wider, more global manner, providing a tool for any user who is for example, doing research on a specific theme or category.

Finally, the last option consists of a combination of the other two searching methods and is the equivalent of a typical advanced search through the archive. While the keyword search provides an easy way of inserting any desired search word, it can be rather inefficient when the user intends to input multiple tag words, whereas the tag selection is limited to this same feature but lacks the flexibility of the keyword search. It is with the purpose of providing such combination of options that this advanced search gathers the information from both inputs and returns the resulting contents as shown in figure 3.8. This example represents the result of an advanced search with both multiple tag words and one keyword inserted, displayed in a table, which is the first of the two visualization options available.



Title	Search Results	Creator
Biography Text	Born in Lisbon, Rui Horta started to dance at the age of 17 at the Ballet Gulbenkian. He ... awards communication dance digital media ...	Rui Horta
Le Bio	the bio of nothing. tags: digital media ...	pmartins
Creation Tool Video Example	... teste_joao.xml tags: digital media war ...	Rui Horta
a slide of images	Images: tags: digital media ...	pmartins
videoplayer	tags: digital media ...	Yaniel
something	tags: digital media Content Type: Other ...	pmartins
Labs Gallery	... Do not show title and description tags: dance ...	Rui Horta
SetUp Annotated Part 1	... argument awards communication dance ...	Rui Horta

Figure 3.8: Tag and Keyword Search - Table

In the previous section it was defined how the navigation of the individual archives allowed the possibility to browse through related contents of the selected items. By selecting such option, the user is redirected to the search interface described earlier with the search parameters filled in corresponding to the properties of the selected content. This

will retrieve a set of content items related to it, where they all share at least one common property with the central content. The related content results are displayed by default as a graph, identical to the one described for the global archive, in order to give the user a quick overview of which contents are closely related to the selected item.

In figure 3.9 are represented the results of a search for contents related to the highlighted item. Once again, the links between nodes represent the connection between them, in this case defined by the tag words contained. As this is a related content search, it is natural that the original content's node has the largest number of links and assumes a central position in the graph itself, as seen in the previous example. The proximity of each content to the central one can easily be identified by the size of the corresponding node circle, which is proportional to the amount of tags it has in common to the ones in the related search.

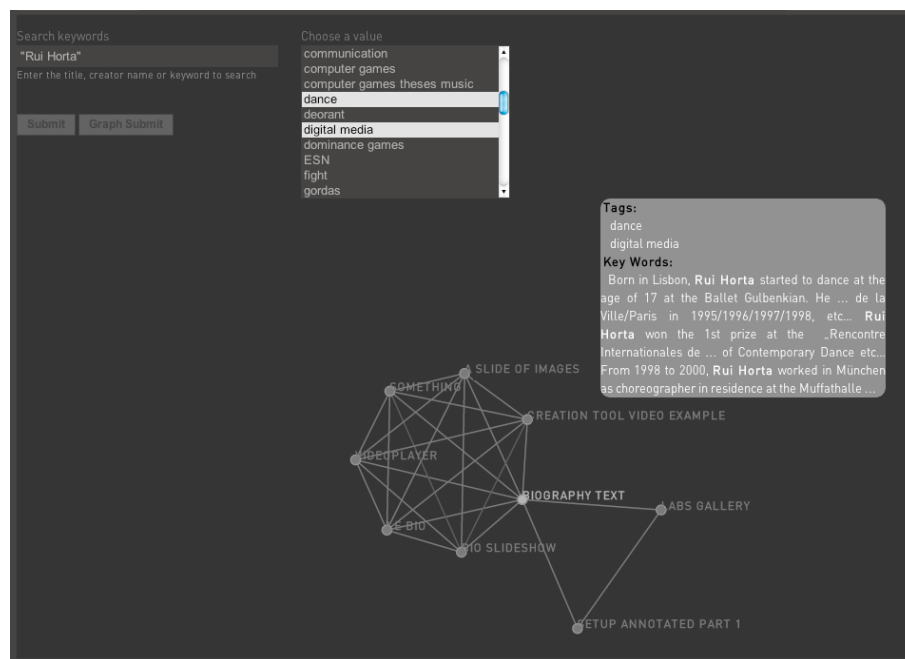


Figure 3.9: Related Content - Graph

Since the related content functionality shares the same base as the normal and advanced search tools, the results visualization modes are the same and can easily be alternated between them at the users will. On both normal and related search it is possible to know how the content nodes relate and which tag words they contain by hovering either through the nodes or links, just like the global archive. However, when key words are inserted, there is the need to distinguish this from the tags. Consequently, both on the table and graph modes, there is a small reference to the source of the relation referencing if the node relates by the tags, key words or both.

Finally, similarly to the functionality of the global archive's navigation, selecting any of the graph nodes or title links in the table will redirect the user to the corresponding content in linear mode.

3.4 Summary

In this chapter we present an overview of the overall project context and its functionalities. We started by reviewing the archive's supported contents, followed by its overall structure and how it is created. Then we describe how the relational properties of the archive's content can be used to display and navigate through them, as well as how they were adapted for search results visualization.

4

Features details and implementation

In previous chapters was detailed the overall workflow of the archive, how it is organized and what are its features and functionalities. These functionalities can be divided into three distinct categories, each of them with specific implementation characteristics:

- Backoffice and content creation
- Relational navigation and content hierarchy
- Video annotations and media content visualization

User and content administration make part of the foundations of the entire system and are the most important aspect to have in consideration when implementing such platform. Therefore, as will be further detailed in the following sections, the decisions made regarding which technologies and approaches to use in the development of the archive's backoffice were focused on its stability and reliability regarding content management, while not restraining the rest of the system's functionalities in terms of flexibility nor the users customization options.

As was previously mentioned, the relational navigation throughout the global and individual archives is done in a similar way, even though the source of the links between each content item are not the same. This reflects on how the system handles the distinct sources of such connections and is able to translate it to the common navigational tool in order to build the navigational system.

The main aspect of a multimedia archiving system is obviously the visualization of such multimedia content. One must consider the fact that the contents may be of various types

and originate from various users, which highly increases the number of content available for public visualization. Therefore, the attention in implementation goes for the efficiency of the system when handling such factors, and especially supporting the features requisites such as the display of annotations during video playback.

4.1 System implementation decisions

When implementing such system it was necessary to keep in focus the fact that the proposed solution must be able to manage a great amount of multimedia content of different types, uploaded by different users and accessed by many more, whilst still maintaining a reliable and secure management of both the contents uploaded and the overall access permissions. Developing this from the beginning is a task which, in order to provide the necessary requirements and functionalities, would require the expense of a considerable amount of time, perhaps unnecessarily, since there are many platforms available which provide a foundation for the construction of a system such as this.

In this case the platform is designated Content Management System (CMS) and Drupal specifically was the one chosen for development. It provides a vast number of implemented base features necessary to build the type of system intended, where all the backoffice functions are done with resource to a graphic interface, providing the easy manipulation of aspects such as user and content administration. Generally, all CMSs share the same advantages when compared to the alternative of programming a system from ground up, where some of the most relevant are:

- Stable and secure core implementation;
- Efficient content management for multiple authors;
- Customizable interface and functionalities;
- Compatibility with specific technologies and code libraries (HTML5 and JQuery);
- Large community support and contribution.

One of the concerns was to assure that the choice of a CMS would not in any way limit the implementation of all the intended features, particularly regarding multimedia visualization. As will be further detailed, both the relational navigation and the video annotation visualization are highly dynamic contents, requiring a large number of transitions and transformations to the display area happening in real time. This archive, like any other web based system, displays its content using HTML, giving the possibility to use Javascript in order to produce such results. Among the available opensource CMSs, Drupal, Joomla and Wordpress stand out from the rest as the most stable, flexible and user

	Wordpress	Joomla	Drupal
Learning Curve	Small: no technical expertise required, simple websites created out of the box	Intermediate: easy installation; more complex websites with experience	High: easy initial configuration but extending the system requires a lot of experience
Administration	User-oriented administration, very easy and intuitive	Graphical and intuitive	Complex set of available operations, not very user friendly
Extensibility	Thousands of plugins available	Very high number of extensions, not very flexible code	Thousands of core and contributed modules and flexible code
Themeability	Easily themeable, large number of available themes	Decent selection of themes, simple extension	Decent selection of themes, complex and granular theming of site elements
Open Source	Yes	Yes	Yes
Technologies	PHP, MySQL	PHP, MySQL	PHP, MySQL
Documentation	Very large collection of tutorials and API documentation	Documentation and tutorials well categorized to various user profiles, API not well documented	Large, well-documented API, as well as dozens of books written about the latest releases
Support	Very big developer community and third party commercial support available	Smaller developer community but more active and accessible	Very big and active developer community behind it
Summary	Simple and easy to use overall.	Not as simple as Wordpress, not as extensible as Drupal	Complex developer-oriented platform, very powerful and scalable.
What is it good for?	Perfect for small sites and blogs.	Bigger but not overly complex sites	Large, interactive sites, with complex operations.

Table 4.1: CMS comparison - information extracted from [16, 14, 10, 5, 12]

friendly. The comparison illustrated in figure 4.1 was produced from various sources researched, in order to reach a decision on which CMS to use.

Even though at first Wordpress, or even Joomla might look like a better option, by providing easy site administration and a much less steeper learning curve, the verdict is that none of these two might have the flexibility of implementation necessary to match our system's specifications. Moreover, while the user support base behind these CMSs might be rather active and helpful, most of these users are to work on what these systems are suited for: smaller websites with little complexity. This might have proven to be an obstacle when faced with implementation issues, which does not happen at all in Drupal since it is a developer-oriented content management system.

Having native support for Javascript and specifically jQuery, together with the high extensibility with custom and existing code, Drupal has almost no limitations as to what features it can support, becoming the strongest option for the core implementation. As mentioned earlier, the implementation of some of the multimedia features of the archive required the use of a technology capable of handling a high number of animations, together with video and audio playback. Adobe Flash was highly suitable candidate for these tasks. However, combining video playback, annotation overlay and timeline animations, along with its integration with the Drupal system, proved to be a much more complex and heavier solution than the one presented in this work.

Considering the factors already described, together with the constant improvement of web technologies led to the use of two of the most adequate for the archive's requirements. The new elements introduced by HTML5, in this case **<video>**, **<audio>** and **<canvas>**, provide an efficient and lightweight solution for multimedia playback with

overlaying annotation drawings, as will be detailed in [4.2.2](#). Moreover, the fact that HTML is the native output format of Drupal's content and it contains extensive support to the HTML5 format was probably the decisive factor of choice, seen as it will hardly limit any of the specified functionalities of the archive.

Finally, and once again due to performance and ease of integration with the other technologies used by the system, the relational navigation using graph drawing mechanisms is implemented in Javascript. With the large number of freely distributed Javascript libraries designed specifically for graph representation, it was just a question of which one would better meet the archive's specifications. As will be detailed in section [4.2.3](#), *d3.js* was the library chosen for this task due mostly to the ease of implementation of the archive's specified functionalities, detailed documentation, extensive list of example code as well as an active community of contributors.

4.2 Technology

In this section, an overview of the main technologies chosen for the development of the archive is presented. Starting with the core of the system, section [4.2.1](#) provides an overview of the main features of Drupal along with the necessary details of its internals. Section [4.2.2](#) analyses the HTML5 specification, the relevant elements used and how it benefits the overall system and finally, an introduction is given to the *d3.js* library in section [4.2.3](#). Note that the following represent only a short description of the technologies, which will be thoroughly detailed in further sections.

4.2.1 Drupal CMS

It is in the effort to balance between efficiency and simplicity of Web design and development that Drupal excels. Basic Drupal websites contains all of the default functionalities of a web-based content management system. Users can easily create, preview and publish content which visitors will be able to view afterwards, while site administrators can manage overall configuration, content management as well as permission levels of each user. The system has a highly modular approach, focused on extensibility and collaboration, where additional functionality is implemented by overriding the core or by adding separate custom modules, meaning that anyone can safely create their own personal features for it without the danger of affecting the system or its existing contents.

Having such flexibility in terms of customization and the fact that almost every core functionality can be intercepted and altered through custom modules or themes takes Drupal to a next level of Content Management System (CMS), often described as a Content Management Framework. While the core functionalities have already been developed and

are ready to use without further configurations, developers have at their disposal an extensive and well documented API which allows them to easily extend their system to suit their needs and preferences.

4.2.1.1 Drupal architecture

Drupal is designed with the goal of reaching as broader audience as possible, where the system should be able to run on low budget servers or hosting companies, as well as scaling up to larger distributed sites. Doing so involves using the most commonly used technologies together with organized and careful core programming. The important technologies used by the system and the corresponding technology stack is illustrated in figure 4.1

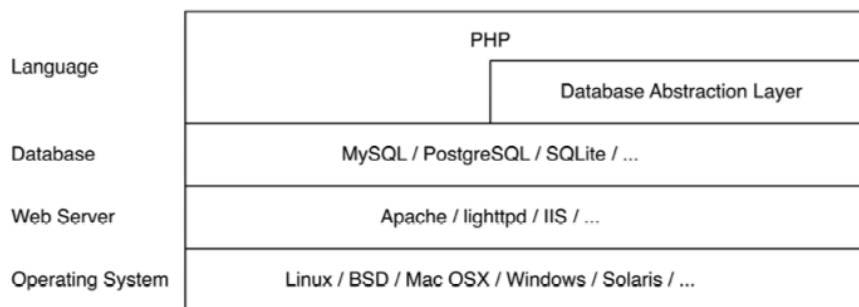


Figure 4.1: Drupal Technologies, in *Pro Drupal 7 Development*[32]

Considering that the operating system is at such a low level of the technology stack, Drupal does not have any special regard to it, running successfully in basically any operating system with minimal requirements. Drupal was originally built only supporting Apache and it is still the most widely used web server together with it, even though other web servers may be used in alternative. One of the advantages of using Apache is that Drupal natively supports *Clean URLs* through the use of Apache's **mod_rewrite** component, which doesn't happen with other web servers.

The database in Drupal is used to store most of the content and configuration settings for the system, while some content such as media files are stored in the server's filesystem. Similarly to the web server, Drupal only supported MySQL in the beginning. However, in latest releases, the database layer connects to the Drupal system through a lightweight intermediate database abstraction layer, providing an API based on PHP data object (PDO), which allows it to support any database that can handle PHP, such as PostgreSQL, SQLite and Oracle.

At the top layer is the Drupal core system and all the contributed modules written in PHP.

The easy learning curve of PHP, which is already one of the most popular web scripting languages worldwide, means that there is less difficulty for new developers to start out their own code and contributing for Drupal's core.

4.2.1.2 How it works

A Drupal installation has the core as the foundation layer of the system. It consists of a number of core libraries and modules which together are responsible for providing the basic functionality such as common functions, as well as support other parts of the system. The core libraries provide functions and services used for low level processing such as database interaction, encoding data or language translation. However, these tools are not responsible for handling the lifecycle of the site's requests. This is where Drupal core makes use of its modular design, giving the modules the opportunity to handle parts of the requests at the appropriate time.

Hooks are probably the most important part of the Drupal system. They can be seen as common callback events although, because their construction follows a function naming convention and not by registering an event listener, they are not being called back but specifically, in Drupal language, they hook into the lifecycle of the request. By following the specific naming convention of "module_hook", where "module" is the name of the custom module and "hook" the hook's system name, it is possible to tap into the core functionalities of the system. Whenever a request is made, Drupal will search through all the existing modules for functions matching this pattern, giving them the possibility to alter the workflow or add functionalities to the system.

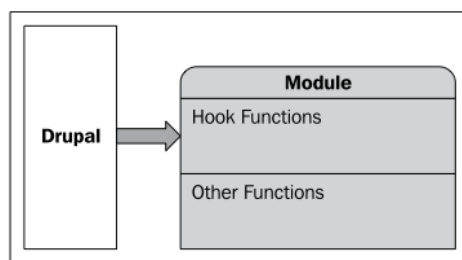


Figure 4.2: Drupal Hook System, in *Drupal 7 Module Development*[18]

The content types defined in section 3.2.1 are all derived from a single base type defined in Drupal as **Node**. Implemented mostly in the node module, it provides a set of hooks meaning that other modules can also interact with it via hook function implementations. This proves useful as a developer may implement features such as ratings, comments, geolocation, among others, for nodes in general, regardless of its type. These generic

features can later be enabled or disabled specifically for each content type by the site administrator to better suit the configuration of the system.

The content of each node can be displayed separately in any part of the page. Blocks are areas defined in the site to display specific content which can be reused in any page intended and, as any other generic element of the system, it is possible to specify which blocks are visible at a given time or for a set of user types. Drupal core and modules are responsible for assembling and delivering the text based content to be displayed. The content's markup can be manipulated not only by these, but as well by the **theme layer**. This subsystem is designed specifically for generating the HTML that suits the content, which will later be received in the browser.

Theming the actual output of the content can be done simply by using a cascading style sheet (CSS) in order to override the Drupal's default element classes and IDs. However, it is easy to customize the actual HTML output, by implementing a custom theme. Each template file of the theme consists of standard HTML and PHP code and additionally, any dynamic part of a page such as a breadcrumb trail, can be altered by overriding the appropriate functions.

4.2.2 HTML5

The specifications of HTML5 bring many improvements to the development of next generation web sites and aim for future unification and multiplatform support of interactive content. The large set of specifications[24] introduced by HTML5 can be divided into the following categories:

- Global page markup
- Visual presentation
- Multimedia support
- Javascript improvement

Among those introduced in HTML5, some of these new elements, such as <header>, <footer>, <section> and <article>, have the goal of transforming the current page markup structure composed of <div> tags into a more semantic system. Although this is not the focus of this work it is good to note that the Drupal system, specifically the theme subsystem used in the development of the archive makes extensive use of such markup elements together CSS3 styling properties, taking full advantage of its adaptive layout characteristics and multi-platform support.

As mentioned earlier, the rich media support of its specification is what led to the choice of HTML5 for the implementation of the archive's multimedia functionalities. Both the

`<video>` and `<audio>` elements are fully integrated as part of the site's structure, just as any other `<div>`, and can be manipulated as such. Through the use of HTML layout positioning and CSS rules, it is possible to have a video as an dynamic structure, which can adapt to screen size changes and is subject to document manipulation just as any other. While the media playback is less heavy than with the use of an external plugin, there is the drawback of the amount media formats already supported. Since HTML5 is relatively new it hasn't reached the level of support given by distinct browsers which we can see, for example, in Adobe Flash Player. The following list, illustrated in figure 4.3, demonstrates the current browser support for the different video formats available:

Browser	MP4	WebM	Ogg
Internet Explorer 9	YES	NO	NO
Firefox 4.0	NO	YES	YES
Google Chrome 6	YES	YES	YES
Apple Safari 5	YES	NO	NO
Opera 10.6	NO	YES	YES

Figure 4.3: HTML5 Video formats, in *W3Schools HTML5 Video Reference* [15]

Although this is one of the major challenges for the adoption of HTML5's specification throughout the most popular web browsers, it has already reached a point where it is viable to implement such elements and, due to its fallback capabilities, if either the browser or the format is not suitable, there is always the possibility to make use of external plugins in order to maintain media playback functionality.

From the set of new HTML5 elements, the one which is particularly useful for the archive implementation is `<canvas>`. This element provides an API for 2D drawing which natively integrates with the site's structure, and just like any other it can be manipulated in any way required. Besides general integration, it has specific compatibility with the `<video>` element, allowing video content to be loaded directly inside it and thus being part of the canvas itself. This allows the use of the array of functions available for the canvas on a video element, providing even more possibilities for its manipulation. These elements' properties and functionalities fit perfectly as a solution for the archive's video annotation visualization. Moreover, as mentioned earlier, the fact that Drupal displays content through HTML and has support for such elements, makes it a simple task to integrate both technologies without the use of external plugins.

4.2.3 Data-Driven Documents - d3.js

Data-Driven Documents (d3.js) is the name given to this Javascript library created for the efficient manipulation of web documents based on data. Rather than providing a framework for data visualization, D3 makes use of the standard Document Object Model (DOM) representation enabling direct and efficient manipulation of its elements, with no additional abstract visualization layers.

D3 has access to the full extent of web standards such as HTML5, SVG and CSS, allowing extreme flexibility for the developers. By binding arbitrary data to an element it is possible to apply any number of data-driven transformations to the document. In figure 4.4 it is possible to see an hierarchical data structure where the nodes and links are composed of SVG elements, which can be dynamically manipulated by D3. As illustrated in this example, there is a large range of display customization available which are totally independent of the data structure itself. This independence between data and its representation makes it a truly efficient tool for varied data presentation, giving the possibility to reuse the exact same data to create, for example, an HTML table as to transform it into an SVG bar chart.

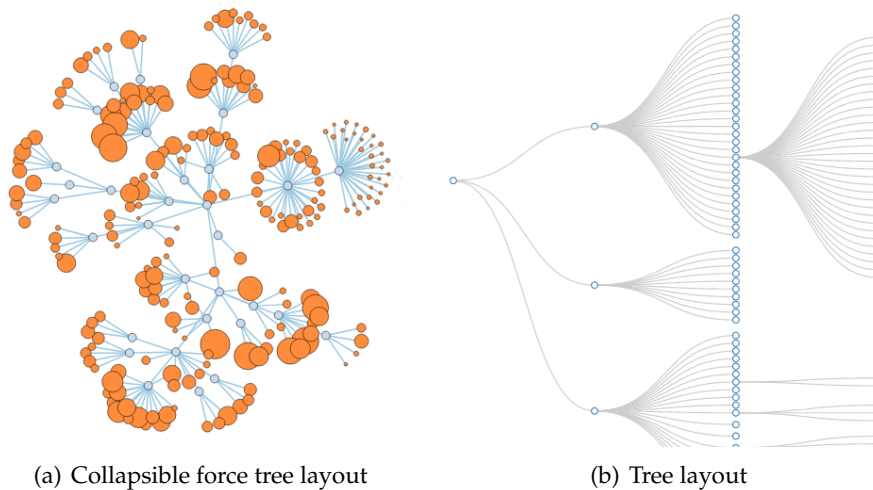


Figure 4.4: D3 hierarchical layouts, in *Data-Driven Documents*[4]

Document elements selection and manipulation with D3 do not follow the imperative approach of the standard W3C Selectors API ¹, which requires manual iteration of each object in a set. Instead, similarly to *jQuery* ² or *Prototype* ³ on some levels, D3 makes use of a declarative approach, applying the desired manipulations to arbitrary sets of objects named *selections*. This way, it is possible to simplify complex operations into simple, natural expressions as shown in listing 4.1. In this example we are selecting all the existing "p" elements in the document and assigning each one a specific *font-size*. The value *d* returned by the *style()* function corresponds to each of the elements in the array passed to *data()*.

¹ `d3.selectAll("p")`

¹ <http://www.w3.org/DOM/DOMTR>

² <http://docs.jquery.com/>

³ <http://api.prototypejs.org/>

```
2 .data([4, 8, 15, 16, 23, 42])  
3 .style("font-size", function(d) { return d + "px"; });
```

Listing 4.1: Node creation

The data to append only has the requirement of being an array of data elements. Moreover, D3 supports the loading of data from external sources such as JSON objects, XML or CSV files, which can be imported and bound to a DOM element. In order to handle more complex data representation, D3 includes a number of predefined element layouts, such as *Tree*, *Pie Chart* or *Force*, which automatically arranges the data-bound elements according to it. The graphs in figure 4.4 illustrate an hierarchical data structure represented in both a force-directed layout and a simple tree layout. As will be detailed in further sections, the graph created for the relational navigation of this archive has force-directed layout, although manually set to represent its tree-like hierarchy.

4.3 Backoffice

As mentioned in previous sections, one of the main reasons for the choice of a CMS to serve as a foundation to the archive is the fact that it provides such stable user and content management functionalities implemented in its core. Regardless of being stable and embedded in the core of the system, the administration and backoffice features of Drupal can be easily overridden and customized to fit the requirements of the archive through the implementation of custom modules and themes. In order to implement the required features of the archive there was the need to customize almost every detail of how Drupal handles and displays content administration. This required the creation of a custom module and theme, providing the possibility to hook into every core functionality and customize it in the intended way, as detailed in section 4.2.1. The module, *tkb_module.module* consists of a PHP file with support for the Drupal API, which can override and add custom handling for all the data to be displayed to the user and stored in the system. This includes manipulation of forms, custom callback functions, page redirects as well as database access which, as will be detailed in the following sections, all have essential roles in building the custom content administration.

With the functionalities being mostly handled by the module and the data properly handled, there is also the need to customize its display and the theme layer is the one responsible for such tasks. The theme consists of a primary file, the *template.php*, combined with a number individual template files such as *node.tpl.php* as well as various auxiliary CSS and Javascript files. The first one allows the creation of preprocess functions specific for system elements which are to be displayed, from an entire page to a specific form or even the breadcrumb trail, it is possible do intercept and customize the markup of virtually everything in the website. Template files are the final piece of the theme layer, providing

a last chance to customize the final markup of any element. Consisting of PHP files, their goal of is to adjust the HTML markup of each element of a page individually, accessing to the data passed by the primary *template.php* in order to print it for the final display. Additionally it is possible to include external scripts and styles to apply to the elements, extending even more the customization possibilities.

4.3.1 Drupal backoffice and content creation

Drupal provides a rather large set of administration tools regarding both user and content management which enables us to easily configure some of the backoffice requirements of our system. The implementation of Drupal's core content type creation and user permission enabled us to easily configure the initial templates, as defined in section 3.2.1, with the necessary fields and appropriate user permissions. This mechanism makes it considerably easy for a site administrator to make changes and further customize the system. However, the content creation itself might not be so natural for an inexperienced user. Since the archive allows any authenticated user to create his own content and, considering the specific target users it aims for, it was essential to transform this mechanism into something accessible for any type of user.

There is no clear separation between content presentation and backoffice, meaning that there is no specific area where all the content administration is done for a specific user. When viewing a piece of content, if the user has permission over it, an "Edit" option will appear allowing changes to be made for that particular item. By default Drupal supports multiple user content creation and permission handling. However, it does not have the concept of individual archive intended for this system and as such, it does not reflect a native separation of content according to its owner. In order to create such functionalities, some major changes were made to the workflow and aspect of content creation and editing for the archive, resulting in individual backoffice areas for each user. The focus behind the implementation of this feature was on providing the user with a natural tool for managing the content of his archive. This requires displaying an overview of the archive's structure, represented by the sidebar in figure 4.5, guiding the user on where to place new content.

Technically, the overall backoffice area represented in figure 4.5 is implemented as a content item of a custom content type designated *custom_page*. This unique page, shared by every user, is distinguished only by the items visible in the sidebar, which correspond to the content owned by the current user. The details on this functionality are part of another topic related to content hierarchy and organization, which is further addressed in section 4.4.1. In order to transform the content creation workflow into the natural 3-step

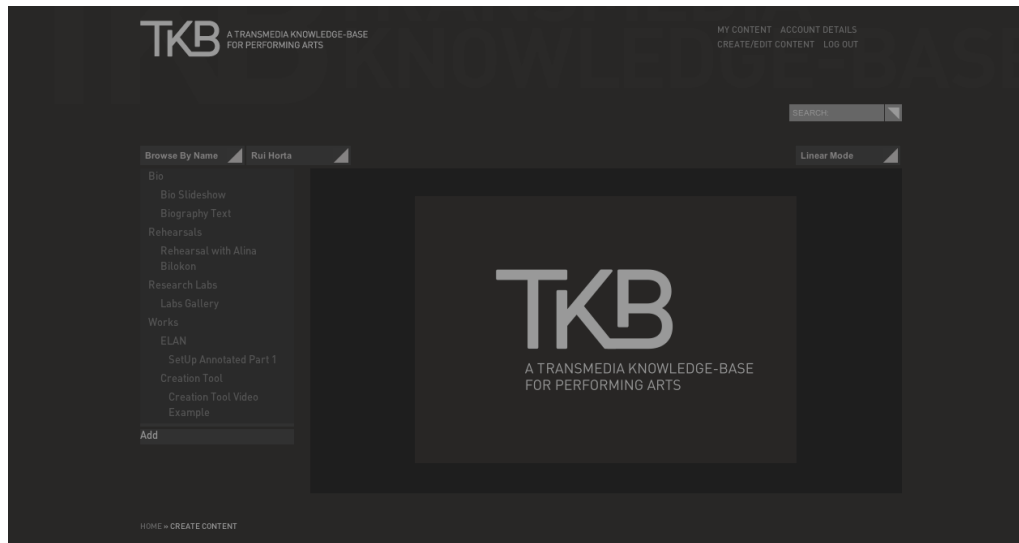


Figure 4.5: Backoffice Area

process described earlier it was necessary to drastically change the structure of the existing forms and how they are displayed. Most of the modifications are done in the custom module, by implementing the *hook_form_alter* function, whose arguments give access to all the elements in the form before it is built for display. By checking the *form_id* it is possible to identify which form is being processed at the moment, and proceed with the respective modifications. Firstly the form fields which are not required to be visible for the user are hidden by setting the appropriate form field's *access* property to false. This does not remove the field from the form but rather hides it from the user, being saved with the default values or custom ones, if defined in this function.

The next step is regarding the form action buttons which need to be re-positioned. In order to do such thing while maintaining normal form functionality it is necessary that the form actions are defined and correctly returned from the module. Therefore, contrary to some of the other fields manipulated, this is done in the *template.php* file as it is part of a visual modification rather than a logical one. Similarly to the module function, *template_preprocess* allows modifications to be made to the form element before the final data is sent for markup.

Finally, after all the logical manipulation is done and the form data is ready to be displayed, it is up to the theme layer to specify the final markup to display. Note that this is a particular example for form manipulation and customization, although the process is relatively the same throughout the rest of Drupal elements. As for content creation, the design specified implements an accordion menu system as illustrated in figure 4.6, which requires the use of an external Javascript library part of the *jQuery* package named *jQuery Accordion*⁴. In order to make use of this library, as with any other external script, it has to

⁴<http://jqueryui.com/demos/accordion/>

be called by either the module or the theme at the appropriate time.

The image shows a Drupal form for configuring a menu. On the left, there are standard form fields: 'Title' with an asterisk, 'Tags' with a placeholder 'Enter a comma-separated', and 'Body' with an '(Edit summary)' link. A dark, semi-transparent overlay is positioned over the right side of the form, displaying a 'Menu Settings' panel. This panel contains a 'Menu link title' field, a 'Parent item' dropdown menu currently showing '<Rui Horta>', a 'Meta Description' field, and an 'Add Content' button.

Figure 4.6: Backoffice Menus

In this case the library is loaded in the *hook_form_alter()* function of the module, being available to be used in every form processed by the system. The first line of the code snippet in listing 4.2, *drupal_add_library*, represents the default loading of the library, while the second one, *drupal_add_js* is used to configure the Javascript object according to our HTML markup. This function call consists of a simple Javascript call to the accordion's object initializer, configured with a specific identifier which will correspond to the one defined in the template file. Therefore, as mentioned earlier, the theme layer must implement a specific template file in order to customize the data markup, in this case following the HTML requirements needed to apply the accordion object. The template file is named *template-node-form.tpl.php* following Drupal's naming convention, and it is there we can define the necessary HTML markup, reorder the form elements and specify element identifiers matching the ones used in the Javascript code, resulting in the desired output and functionalities.

```

1 drupal_add_library('system', 'ui.accordion');
2 drupal_add_js('jQuery(document).ready(function() {
3   jQuery("#accordion").accordion({
4       collapsible: true,
5       active: false,
6       autoHeight: false});});', 'inline');
```

Listing 4.2: Accordion configuration

Since Drupal maintains this "always-on" content editing approach described earlier, every time a content form is submitted regardless of the action, the current page will redirect to the content visualization mode, exiting our so-called backoffice. Therefore, one of the implementation details that was essential for the creation of the specified backoffice

workflow was the interception of page redirections regarding content form submissions. Similarly to the form manipulations already described, intercepting a page redirect consists in the modification of a specific form field, although in slightly different way. Since this is an action that happens after the form data has been submitted, it is necessary to create an additional callback function for the form submit action, where it is possible to change the form's *redirect* field, in this case to the *custom_page* created for the backoffice area, and thus generating the aforementioned workflow.

There are, however, many more implementation details regarding form handling and submission such as content menu hierarchy, which were not detailed since such functionalities fall into slightly different contexts, which are to be addressed in further sections.

4.3.2 ELAN and Creation Tool annotations parsing and storage

Video annotation has been mentioned in previous chapters as being part of the content types supported by the archive. What happens with these tools is that the information regarding each annotation is exported into a separate file, typically in XML format, with a specific structure which naturally is not natively supported by the core functionalities of the system. Therefore, in order to display the annotation information and provide all the filtering mechanisms described in section 3.2.1, it was necessary to parse the annotation files correspondent to each video, storing all the relevant information for further manipulation. This section describes the implementation of such mechanisms as well as the supporting data structures created for such.

The main goal of this parsing mechanism is to implement customization options for the annotation visualization, namely individual control of visibility and respective category colour code. With resource to the form handling capabilities described in the previous section, it was possible to devise such customization mechanism with the following workflow:

1. Creation of hidden form fields for annotation information
2. Annotation file parsing and customization options
3. Storing customization data in the appropriate structures

Seen that the differences between the annotated content from the two tools are minimal, regarding only on the structure of the annotation document, it is essential to note that the process is practically identical for both of them. Therefore, on account of simplicity, the following will only describe it for the Creation Tool with references to the existing differences.

In order to control the individual annotation visibility and the color for the annotation category it is necessary to store such information. For such purpose the two tables shown in figure 4.7 were created, storing the appropriate info for each annotation. The entries in both *ct_annotations* and *elan_annotations* are respective to each annotation, identified by the owner of the content, *uid*, the correspondent annotation file id *fid* and an unique id in the form *category_id-annotation_id*. The main differences between the CT and ELAN tables is the existence of the *type* field in the first identifying the annotation type, and the *contents* field in the latter, which may hold the annotation content for future system searches.

ct_annotations	
uid	"user id"
fid	"file id"
id	"anno id"
type	"anno type"
visible	"visibility"
color	"hex color code"

ct_colors	
uid	"user id"
fid	"file id"
ink	"color"
def_ink	"default color"
...	...

Figure 4.7: Annotation database tables

The main color tables, *ct_colors* and *elan_colors* have a slightly different structure, due to the fact that CT has a fixed number of annotation types, while ELAN's categories are defined by the user while annotating the video. Therefore, in the first table there is one unique entry for each *uid*, *fid* pair, with color information regarding all the annotation types while in the ELAN table there is an entry for each category specified within the file. In order to provide the intended customization options it was necessary to alter the form structure manually, adding two hidden fields which will be modified by the user through the custom interface. Using the *hook_form_alter* function described in the previous section we add two new text fields to the form element named *color_info* and *vis_info*, which hold respectively the color and visibility information.

Parsing the annotation files is handled by a custom module function, which is only executed when the form "Upload" button of the file field is pressed. The annotation file uploaded is loaded into a readable array by the PHP native *simplexml_load_file* function, which can then be correctly parsed in order to retrieve all the annotation information necessary to build the customization interface illustrated in figure 4.8. Since this is a dynamic functionality, which should only be available when the annotations file has been uploaded, the Javascript code that handles the interface creation is only loaded after all the annotated data has been processed.

By accessing the variables with the annotation information and the respective categories it is possible to generate the HTML markup for the collapsible hierarchy. The "Edit Color" option opens a color picker which allows the color selection for each annotation category



Figure 4.8: Annotation controls

reflected in the content visualization, as mentioned in section 3.2.1. This color picker is loaded from Farbtastic⁵, an external jQuery plugin which provides specific handling functions for the selection, allowing the storage of the color information in the appropriate data structure. The visibility toggle, as the name suggests, allows the user to select which annotations are to be displayed with the video playback. This is handled through a jQuery function bound to the visibility toggle element, which corresponds to a specific annotation. Both mechanisms store the respective data in arrays, *color_array* and *vis_array*, which are only processed when the "Save Changes" button is pressed, passing the annotation information to the string format mentioned earlier in order to alter the custom hidden form fields. Finally, with the customization information stored in the form, the submit callback function will make sure new data is inserted and existing entries are updated in the respective color and annotation tables.

4.4 Content hierarchy and relational navigation

Providing the intended relational navigation of the multimedia archive requires the implementation of the content's inherent relations. The features described in 3.3.1 already illustrate how such relations are represented and the way they allow the relational navigation. Therefore, the goal of the following sections is to detail how these connections are made and what mechanisms are used to both store and display them.

4.4.1 Content hierarchy

Even though the graph representation mechanism is the same, the source behind node connections in the global and individual archive are completely different, which reflects in its implementation. Therefore, on account of simplicity, the individual and global archive's connections are defined as **menu** and **tag** hierarchy respectively and as such they will be analysed separately.

⁵<http://acko.net/blog/farbtastic-jquery-color-picker-plugin/>

4.4.1.1 Menu hierarchy

As mentioned previously, in Drupal, contents are connected to the user who creates them. However, there is no separation as to where they are stored. Moreover, the only core mechanism which allows the creation of some sort of hierarchy between content is the *Menu System*, and even this one does not contain all the needed functionalities needed to implement the required features. Therefore, in order to provide such features some further modifications were made to the content creation in the backoffice, in addition to the ones described earlier.

Drupal's core menu system allows the creation of links pointing to the contents created. It is possible to define different menus for distinct purposes and the site administrator may select which one to use for the various content types. In a closed system where there is a small, predefined number of users, it would be possible to create individual menus for each one. For this system, even though it would be possible to create a new menu for every new user, it could lead to complexity problems as its number may escalate quickly. Therefore, the solution implemented uses a single menu for the content navigation of all the users, filtering out the unnecessary links.

When creating new content the user is presented with a representation of his archive, illustrated in figure 4.9 as the links in the sidebar (1). Since, technically, it is only a custom representation of pre-existing menu element, the manipulation is done in the theme layer and the following functions specifically, in the *template.php* file. Retrieving menu information is done through existing core functions. In this case, since we require all the levels of the menu hierarchy and its details, it is used the *menu_tree_all_data* with further modifications to the returned data since it does not consider the user filtering mentioned.

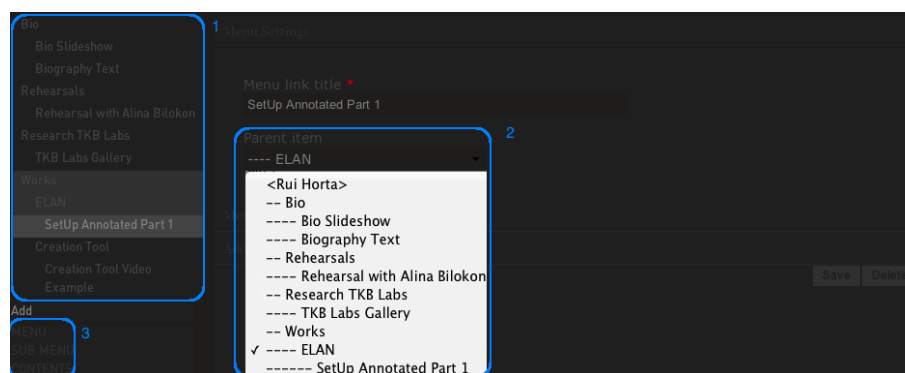


Figure 4.9: Menu items

After retrieving the menu data, it is also necessary to gather information about all the

current user's previously created content. This is done through a simple database access, retrieving all the links and ids of the respective nodes, which will be used together with the menu data in the custom `_process_menu_tree` function, represented in listing 4.3. This recursive function makes use of the retrieved user content information, generating a custom menu tree hierarchy where all the links which do not belong to the current user are filtered out. As detailed before, this custom data is passed to the custom template file for the sidebar's block, which takes care of printing the HTML markup in the tree hierarchy presented in figure 4.9.

When parsing the custom menu data in the template file it is possible to create custom links for each item, passing as variables the respective level and content type. The *level* variable passed in the link represents the depth of a specific menu item, which is used to control whether or not it is possible to create more content under it, reflected in the menu buttons (3) in the bottom of the sidebar which are highlighted or not accordingly.

```
1 function _process_menu_tree(menu_tree, user_links, level)
2   forall link in menu_tree
3     if link in user_links and link has children
4       childs <- _process_menu_tree(children, user_links, level)
5       new_menu_links <- array{title, link, level, childs}
6     else
7       new_menu_links <- array{title, link, level, childs}
8   return new_menu_links
```

Listing 4.3: Custom menu creation

In accordance to the menu structure in the sidebar the user is presented with a list of all his previously created content illustrated in (2). This list allows the selection of the parent menu item for a newly created item, as well as reposition existing ones, always regarding the restrictions mentioned earlier. Unlike the sidebar, this menu item selection is part of Drupal's core functionality, embedded in the content creation form, although it shares the same drawbacks regarding user content separation. Thus, since it is necessary to filter the unwanted items and the list is part of the form, this process is done in the custom module and, once again, through the *hook_form_alter* function. The custom code created implements *_build_custom_parent_menu_options*, a recursive function almost entirely extracted from the core *Menu* module with the necessary modifications in order to select only the current user's content. Since the structure of this function is almost identical to 4.3 therefore there is no need to further detail its implementation. Note that, by selecting a link on the sidebar before proceeding to the content creation will predefine it as the new content's parent item. This is done by passing a menu item id through the link which is then set as the default value of the form field.

4.4.1.2 Tag hierarchy

Describing content through the use of tag words is one of the most simple and natural ways for users to establish connections between them. As mentioned in earlier sections, relating content from distinct users is one of the main requirements of the archive's specification as the graph built from it will serve as the system's landing page, essential for content browsing. In this sense it is a rather large benefit the fact that Drupal has a built-in Taxonomy mechanism, which serves the purpose of the core of our implementation.

The only drawback of the existing tagging mechanism is mostly regarding its interface and the creation of new terms. By default the tags, or vocabulary, are defined in a specific system administration area which is normally managed by users with higher privileges. Words in this vocabulary can later be assigned to a specific content type via a term reference field, which only allows the selection of pre-existing words. With the purpose of extending such mechanism a custom interface was implemented, illustrated in figure 4.10, which allows both the selection of existing words (1) and the creation of new ones (2).

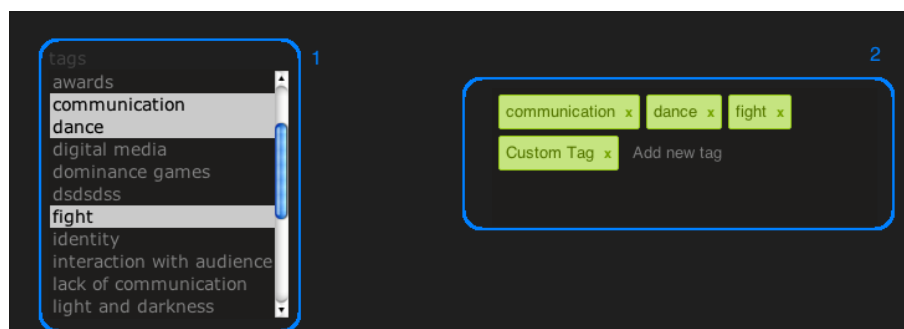


Figure 4.10: Tag creation

The tag creation interface (2) is implemented with an external Javascript library named *XOXCO*⁶, which very simply transforms a form input field into a more interactive and natural element for the purpose intended. Inserting new words will automatically highlight them in the box, as well as append them to the custom form input field. By binding handling functions in a custom script, it is possible to create interaction between the default tag multi-select and the custom insert fields. This means that every word selected in (1) will be part of the input field in (2), making it easier to identify all the words. In the same way, deleting any pre-existing word from the input box will consequently de-select it from the other one.

It is important to note that when creating an intermediate menu structure, as described in section 3.2.2, it is necessary to categorize it, and consequently the contents placed under

⁶<http://xoxco.com/projects/code/taginput/>

it. In order to do this the user is required to choose one category from a second list of options such as *Works*, *Biography* or *Videos* for example, which better categorizes the content to be stored below that item. This field is relevant to the global archive navigation and as such will be further addressed in the following section.

Providing the mechanism to save the words inserted into the existing vocabulary is however, the crucial part of this implementation and, once again, it requires the interception of the form submit. The created function, *add_taxonomy_terms*, accesses the custom form input field in order to parse all words, intercepting those with the ones in the database to confirm that there are no duplicates. Finally, new terms are created with the remaining words according to Drupal standards, which are then saved through a pre-existing core function named *taxonomy_term_save*. Since these tag words are used to categorize individual content items by using a global mechanism, it is easy to discover those who share common words, thus serving as a foundation to the global archive navigation detailed previously.

4.4.2 Graph representation

Previously we defined both the functionalities (3.3.1) and the data necessary (4.4.1) for the construction of the global archive. The goal of this section is, therefore, to detail how such data is transcribed into the graph navigation of the archive and the mechanisms involved in its navigation. The foundation for the archive's global and individual relational navigation relies on the data stored by the mechanisms described in 4.4.1 and therefore it is normal that some of the functions retrieving such data are shared between them and the graph construction.

4.4.2.1 Global archive: tag related graph

The graph representation of the global archive can be done in two ways, distinct only in what the nodes represent: *content* or *users*. In order to support both representations and, since the option is available for the user to switch between them, it is necessary to extract all the corresponding data into two distinct objects, which are then handled correctly by the graph script when requested.

For the user graph, further referred to as main graph, we start by gathering all the nodes with the info from all the users in the system, as well as the taxonomy terms related to each user. As presented in listing 4.4, in order to build the existing links between each node, it is necessary to identify which nodes have the same taxonomy terms associated to them. Moreover, since two nodes may be connected through more than one tag word, we need make sure that these links are not duplicated and instead only one link is created,

associated with all the correspondent tag words.

```

1 function _build_main_page_graph()
2   nodes <- users
3   user_tags <- select terms per user
4   forall user_tags as tags
5     forall tags as tag1
6       forall tags as tag2
7         forall nodes as node
8           if node->userId == tag1->userId
9             source <- node
10          if node->userId == tag2->userId
11            target <- node
12          if tag1 != tag2
13            forall links as link
14              if link->source == source AND link->target == target
15                || link->source == target AND link->target == source
16                link->tags <- tag1
17                duplicate = true
18              if not duplicate
19                links <- array(source, target)
20 graph <- array(nodes,links)
21 return graph

```

Listing 4.4: Main graph creation

Building the content items graph or secondary graph, shares the same process in order to build the links. As mentioned earlier, all content items are categorized by the respective tag words chosen, and even further by inheriting the secondary category chosen to the parent menu item container. This secondary category is used to provide even more alternatives as to how the contents are displayed and as such, the user has the choice of what type of content are to be gathered to create the nodes. In order to do this it is necessary to discover which specific content items are stored under the menu items matching the selected category. This is done once again accessing the menu system's information, through the use of the previously mentioned *menu_tree_all_data*. Using a list of existing content items corresponding to the selected category, it is possible to parse through all the menu tree hierarchy, identifying the intended items and proceed to extract the information regarding all the content stored under them. This results in a list of nodes consisting of all the content categorized by the term selected, which can finally be used with function 4.4 in order to generate the links and the final secondary graph.

With the graph's data correctly build it can be passed to the custom script *global_graph.js*, which handles the its creation and manipulation, including the filtering mechanisms. The initialization of a *D3* graph object with force layout may be done with additional configuration to its parameters, defining how the nodes and links are positioned regarding one another. From the set of available parameters to configure, the focus goes to *gravity* and

charge. The first simulates gravity forces which get stronger as the distance from the centre increases, pushing the nodes to the centre or allowing them to float around freely as the value goes from one to zero respectively. Charge controls the attraction or repulsion of nodes between them, whether the value chosen is positive or negative. In addition to these parameters, it is important to define a custom *tick* function, which handles the specific coordinate positioning of each node and link. This function is triggered in every step of the graph positioning algorithm customizing its final solution. The nodes positioning in the global archive graph is restricted to a specific area within the range $[radius, width - radius]$ for x and $[radius, height - radius]$ for y .

This type of graph layout will run the necessary steps of its algorithm in order to find the optimal solution, considering all parameters configured, stopping eventually to avoid wasting resources. One additional parameter regarding the links which helps speed up the algorithm is *linkDistance*. By setting this, the size of the links will vary according to the nodes current position but it will eventually converge into the specified value, freezing the graph's positioning.

Toggling between different content categories requires rebuilding the graph, assigning the proper node data. This is done by triggering the *init_graph* function which starts by removing the old nodes and links, followed by the assignment of the new graph data according to the category selected. An important detail about the D3 implementation is that, whenever a change occurs to the graph object, there should be a call to the *update* function which handles the insert and removal of new and old items respectively.

```

1   node = svg.selectAll("g.node")
2       .data(nodes.filter(function(d) {
3           return (d.vis==1)}), function(d) { return d.index; });

```

Listing 4.5: Graph node filtering

Node visibility control is required in order to implement the specified graph's filters. Whenever a tag word is toggled through the graph's filter menu the *toggle_tag_visibility* function is triggered in order to handle node and link visibility. By setting a custom field in every node and link objects it is possible to dynamically show or hide any element. This field is controlled regarding each node/link and the number of corresponding tags, which means a node will only be hidden when none of its tags are visible. The code snippet in listing 4.5 illustrates the filtering of node data by verifying the visibility field *d.vis*, which is done in the *update* function whenever a change occurs in the graph.

4.4.2.2 Individual archive: menu hierarchy graph

In section 4.4.1 was mentioned that the menu hierarchy defined would be reflected in the individual archive's structure and as such, the same function, *process_menu_tree*, is used to retrieve the menu data necessary, further used in the custom graph handling script. Before proceeding to the specific graph implementation, it is good to address the secondary navigation mechanism, implemented in form of an horizontal menu bar which follows the graph navigation. This menu specification can be considered as a breadcrumb trail menu and, as such it required specific customization.

The menu data is obtained through the function illustrated in listing 4.6, which uses the complete menu tree hierarchy to find the correct children links for the current path. The function starts by parsing all the first level menu items and comparing its link to the current system path and, if it is the same, the corresponding menu item is returned in order to build the custom menu with it. If this does not happens it means that the intended item is not in the first level, therefore it is necessary to check all the bottom level items. The recursive function *check_child_items* serves exactly this purpose, returning either "-1" if the path was not found, or the corresponding child menu items if succeeded.

```

1  function _build_user_content_nav_menu(nav_menu,nodes)
2      forall item in nav_menu
3          if item->path == current_path()
4              return item
5          else
6              if item has children
7                  childs <- check_child_items()
8                  if childs != -1
9                      return childs

```

Listing 4.6: Custom menu data

The previous mechanism works only for the menu items below the user name. Since the user name is not stored in the system as a menu item it is necessary to build this first level separately. In order to do so it is necessary to first, obtain the list of content created by the intended user and following, the function *_build_user_content_nav_root_item* parses through the first level of the menu tree identifying the items which belong to the user content list, returning the resulting first level of the custom menu. Note that this menu, besides accompanying automatically the relational navigation of the archive, is also consistent with the linear visualization of content, even though the first one is dynamic and the latter static.

Aside from the actual navigation and menu manipulation, creating the graph layout for

the individual archive is not so different from the global archive's. Since it is a force-directed layout as the previous and the graph is already created with the necessary structure, we just need to "force" the tree hierarchy layout. As listing 4.7 illustrates, after the graph is created we reposition the source and target of every link, in opposite directions, to a custom k distance from its calculated place.

```
1 links.forEach(function(d, i) {  
2     d.source.x -= k;  
3     d.target.x += k;  
4 });
```

Listing 4.7: Forced tree layout

The graph's navigation functionality is actually based on graph clustering mechanisms, applied to this tree layout created. Selecting a node in the graph triggers two main actions. The first, detailed in listing 4.8 toggles the visibility of the clicked item child nodes by transforming the hidden nodes array *d._children* into the visible one *d.children* and likewise in the other situation. Following this, some auxiliary functions will update the clicked node in order to create a highlighted "active" path. The second action triggered, function *graphNodeClick*, is regarding the secondary horizontal menu. Having access to all the menu structure it is possible to dynamically manipulate the menu element, deleting or appending levels of the active archive's path.

```
1 function click(d, nav)  
2     if (d.children)  
3         d._children = d.children;  
4         d.children = null;  
5     else  
6         d.children = d._children;  
7         d._children = null;
```

Listing 4.8: Click function

Finally, in order to provide natural transitions between *cloud* and *linear* visualization modes, additional functionality was created in this graph layout. As mentioned in section 3.3.1 there is a additional menu in the bottom of the graph area, allowing said transition between visualization modes. The particularity in its implementation is that, when going from *linear* back to the graph display, there is a handler function, *gotoNode*, which identifies the content and automatically navigates the graph, creating the appropriate active path. This function parses through all the nodes in order to identify the intended content and following applies to it and its parent nodes the aforementioned *click* handler functions, in order to expand the necessary path.

4.4.3 Search

The implementation of the global archive provides an efficient tool for content browsing with some filtering capabilities and, although this may suffice for a large set of users, those looking for more specific items may require additional functionalities. By using the pre-existing system mechanism, properly adapted to the archive's specification we are able to create a custom search tool with such functionalities and custom results display, both linear and relational. Therefore the focus of this section is to detail how such mechanism is implemented and how it links to the rest of the system.

Search parameters in this custom mechanism may be of two distinct sources: **keywords** or **tags** which correspond respectively to a text field, *search_text* and a multi select list of tags, *search_tags*. Although there are two input methods for the parameters, they are merged in order to share the same results callback function. Even though there is the possibility to implement the entire mechanism by hand, Drupal's core search system implements the necessary functionalities handling proper query creation, and as such may be used in order to retrieve the search results. Therefore, the custom implementation has the purpose of correctly merging both input fields to be used by the appropriate function, as well as handling the results, displaying the intended fields in the custom visualization modes.

The tag list is filled through a simple module function named *get_faq_terms* which returns all the available tag words and respective ids of the vocabulary mentioned earlier, used to describe content, while keywords inserted in the form input field will be matched against content titles, user names, text or even tag words in the entire system. As illustrated in listing 4.9, the selected tags are merged into the text input field, which is then passed as argument in the *node_search_execute* core function in order to retrieve the results.

```

1 foreach($form_state['values']['search_tags'] as $key => $value)
2   $res .= $form['search_tags']['#options'][$value]. " OR ";
3
4 $res.=$form_state['values']['search_text'];
5 $text_results = node_search_execute($res);

```

Listing 4.9: Mergin search input

With the result set obtained we proceed to build the customized visualization. As mentioned in section 3.3.3, there are two options to view the results, linear or relational, consequence of the type of search chosen, table or graph, which are handled by the respective results callback function, *custom_search_ajax_result* and *custom_search_ajax_graph_result*. Both functions have the same purpose, using the result set in order to build the display, but while the first one returns a table with the various rows of result items, the latter

builds a graph identical to the main page's. Note that the implementation and manipulation of the results graph is identical to the one described for the global archive and, therefore, there is no need to further analyse it.

One last feature which has also been previously mentioned is the possibility to do a direct related content search. When navigating the individual archive, hovering through a node opens a small options menu illustrated in figure 4.11, which allows either to go the content visualization mode or to do a search for content related to this item. The related content search uses a parameter passed through the link, *rel_id*, which identifies the node intended for the search. If the form handler function detects an id in the link it automatically gathers all the tags associated with such node and triggers a normal graph search with these parameters.

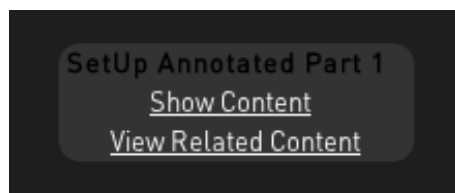


Figure 4.11: Archive context menu

4.5 Annotations and media content visualization

The top layer of the archive's implementation is, obviously, the content visualization. While some of the content types described early do not require any customization besides minor layout adjustments, others rely on unique displays which require an entire custom implementation. In the following sections will be detailed how multimedia content is handled by the system, the modifications made to the core mechanisms as well as the custom ones implemented for the annotated content visualization.

Visualization customization in Drupal is mostly done in the *template.php* file by overriding default values into the intended results. In order to load the required auxiliary files which handle the media display we implement the *theme_preprocess_node* function, and override the individual template files adjusting the final markup details.

4.5.1 Multimedia content types

Most of the simple multimedia contents specified are supported by Drupal with the use of contributed modules. The implementation of inline images and videos, for example, is done with the use of an external WYSIWYG⁷ editor attached to a normal content field.

⁷<http://drupal.org/project/wysiwyg>

This kind of editor provides the user with natural display manipulation of a text field, making it easy to combine it with multimedia fields without the need for any further system customization.

The multimedia gallery template is also one of the content types which did not require additional implementation. Since the used module, *Media Gallery*⁸, is one of the most active in Drupal's developer community, it already provides a wide range of configuration options and customization, drag-and-drop re-arrangement, among others, which does not require us to implement further functionalities.

The remaining content types, however, require the additional manipulation mentioned earlier, in order to create the desired custom visualization, and shall be addressed further in this section.

4.5.1.1 Video player

The video player template is probably the simplest content type, which reflects in the complexity of its implementation. Consisting of a video file, a small text description and the aforementioned tag meta description, the display focuses on the video content itself. In order to simplify video playback in the overall system, we enabled an external Javascript library named *VideoJS*⁹, compatible with the Drupal media system, which makes it easy to create and customize the HTML5 `<video>` elements necessary. This library already provides all the video controls and, in this case, the only required modification is regarding its style. For this purpose we may customize the video markup identifiers by overriding the *videojs.tpl.php* file, which builds the video player HTML elements. Moreover, in order to customize the overall style of the controls, we override the library's default style CSS, making the changes on the *video-js-custom.css* file.

The *VideoJS* library, when enabled in Drupal works for all the intended content types using video fields. Therefore, when implementing the annotated video content types these customizations can be shared between them, leaving only the necessary display specifications left.

4.5.1.2 Slideshow

Similarly to the video player, the slideshow template implementation is quite simple, resorting once again to an external plugin for the slideshow animation. *Orbit*¹⁰ is a

⁸http://drupal.org/project/media_gallery

⁹<http://videojs.com/>

¹⁰<http://www.zurb.com/playground/orbit-jquery-image-slider>

lightweight image slider built with jQuery, which provides all the necessary functionalities and customization possibilities at almost no system resource expense. Configuring the slideshow requires once again adding the plugin's *jquery.orbit.js* and *orbit.css* for the core functionalities, the *custom_slideshow.js* in order to initialize the slideshow object with the intended parameters, such as automatic slide, bullet and arrow navigation, among others. Finally, overriding the *node-slideshow-template.tpl.php* file, we can take the previously uploaded image files and add them within the appropriate HTML element as illustrated in listing 4.10, which will later be used to construct the finalized slideshow object.

```
1 print '<div id="custom_slideshow">'
2   foreach ($content['field_images']['#items'] as $key => $img)
3     print '';
4 print '</div>';
```

Listing 4.10: Slideshow markup

4.5.2 Annotated content

Since the type of annotated video content to be supported by the archive is not commonly used throughout the web, there are no mechanisms already developed which could support the implementation of the necessary features. This lack of pre-existing support materials led to the creation of a unique annotated video visualization tool which, naturally, required a more complex implementation. Although there are differences between ELAN and CT, mostly due to the types of annotations and the correspondent annotation document, the implications on the code are not so extreme. Therefore, the following will detail the implementation specifications in general, with references to the parts with significant differences.

The first step into building the custom annotation visualization is retrieving the necessary annotation data. This is done in the template's *theme_preprocess_node* function, which allows us to query the aforementioned annotation and color tables for the information regarding individual annotations' visibility and color code. The other essential data is the annotation document, loaded by the *simplexml_load_file* function in order to transform the XML file into a readable array object. However, this array must be turned into a JSON object in order to be able to pass it to our Javascript functions. Having the annotations and its color and visibility data passed to our Javascript functions, we may start the construction of the corresponding timeline and filter information.

Before proceeding to the creation of the interface it is good to clarify the initial structure of the template's HTML markup and the differences between both of them. While overall they share the same layout, there are a few key differences which reflect in the

implementation. First of all, since ELAN does not support digital ink annotations, only the CT template uses a `<canvas>` element in order to draw them on top of the video.

AUDIO ANNOTATIONS	MAIN TIERS
TEXT ANNOTATIONS	main tier 1
MARK ANNOTATIONS	main tier 2
LINK ANNOTATIONS	SUB TIERS
	sub tier 1
	sub tier 2

Figure 4.12: Annotation information structure

The most important aspect of the markup, illustrated in figure 4.12, refers to the annotation categories in the different content types, reflecting the hierarchy detailed earlier for the backoffice. Since CT has a pre-defined and immutable number of annotation types we are able to build the full layout structure, where each container only displays annotation from the correspondent annotation type. With ELAN files, however, we are not able to predict the amount of different categories and tiers, and the corresponding hierarchy, therefore we only create separate global containers, into which we can append the tiers and corresponding categories. This is done dynamically in the appropriate Javascript functions when the annotation files are being parsed and shall be addressed further in this section.

When the document and all the necessary files are successfully loaded the *startParsing* function is triggered in the Javascript file. Shown in listing 4.11 are the functions used to, first to read the JSON object passed from Drupal, and then creating the readable array with the *parseJSON* function.

```

1 var xml_file = Drupal.settings.xml;
2 var xml_obj = jQuery.parseJSON(xml_file);
3 parseXml(xml_obj);

```

Listing 4.11: Annotation document

As mentioned in the previous section, the video playback is implemented with the use of the *VideoJS* plugin and as such, we use the provided functions to initialize the video object. Since the creation of the timeline relies on attributes of the video, such as the total length and dimensions, it is necessary to wait for the *video.attr('readyState')* attribute in order to proceed with the remaining operations. The interface provides the possibility to toggle the visibility of both the sidebar and the timeline, resulting in various layout possibilities. Moreover, since videos may have distinct resolutions, we need to dynamically calculate the appropriate width and height for the video, and the matching canvas in the CT template. In order to do this we store the fixed layout values, as well as the video's native resolution, and in the appropriate callback functions, such as listing 4.12, we verify

which layout elements are hidden in order to resize the video and canvas objects correctly.

Using the data regarding annotation visibility we can filter which ones are to be created and appended to the timeline. The *addAnnotation* function is called for each annotation, where all their properties are extracted in order to create the appropriate annotation object and store them in the necessary data structures. Creation Tool annotations, besides having the time frame reference, contain different attributes regarding their annotation type: **ink** has a list of x and y coordinates and thickness, **audio** and **mark** have the path to the respective files, and so on. ELAN on the other hand is simpler since the only type of annotation is text, containing only the time frame, tier identifier and annotation content. Moreover, with the information regarding the specific beginning and ending time slots of each annotation we are able to calculate the appropriate square size to append to our timeline.

```

1      $("#vid_bot").toggle(showOrHide);
2      if(showOrHide)
3          myPlayer.height(fixed_video_height);
4          $(".ct_tiers_anno_info").css({'max-height': 321});
5          if(toggleAnnoVis)//this means that the sidebar is also hidden
6              resizeVideoElements(921,fixed_video_height,false);
7          else
8              resizeVideoElements(fixed_video_width,fixed_video_height,false);
9      else
10         myPlayer.height(571);
11         $(".ct_tiers_anno_info").css({'max-height': 570});
12         if(toggleAnnoVis)//this means that the sidebar is also hidden
13             resizeVideoElements(921,571,false);
14         else
15             resizeVideoElements(fixed_video_width,571,false);
16         showOrHide = !showOrHide;
17         for(var i = 0; i < num_anno; i++)
18             annotations[i].focus = false;

```

Listing 4.12: Toggle video size

Creation Tool annotation files have time referenced in frames and as such we are assigning half a pixel for every frame the annotation lasts, $(end - begin) * 0.5$, resulting in an acceptable square size in the timeline. However, the ELAN annotations are represented in seconds and thus we need to adjust the square size calculation. Considering an average value of thirty frames per second we adjust the calculation, maintaining the same idea behind it resulting in $((end-begin)*30*0.5)/1000$. Note that, by not limiting the annotation square sizes to a specific width, we will have some of them positioned outside of the display area. Therefore, in order to solve this issue, we implemented a scrolling mechanism, which automatically moves the timeline showing the hidden annotations. Listing 4.13 illustrates the manipulation of the timeline at each time update call. As already mentioned

we are assigning half a pixel for every frame and, since the HTML5 video's *currentTime* function returns the current position of the video playback in seconds, we need to adjust this value in order to compare it to the length of the timeline element. If the current time position exceeds half of the timeline we use jQuery to move the timeline and the auxiliary seek line.

The function in listing 4.13 also triggers a custom *checkAnnotationRange* function, in order to display the stored annotations at the proper playback time. It takes the current time passed by the *timeUpdate*, parses through all the annotations stored which are to be visible and checks the corresponding start and ending time. If the current time is within an annotations time slot, this annotation is highlighted and the *writeAnnotation* function is called, handling the annotation visualization.

```

1  if((now * 30) / 2 >= 795 / 2 & scroll_timeline)
2      if(old_seek < middle_point)
3          old_seek = middle_point;
4          jQuery("#seek_line").css({'left' : middle_point});
5          aux_seek = (now * 30) / 2;
6          jQuery(".anno_wrapper")
7              .css({'left' : parseInt(old_pos) - (aux_seek - parseInt(old_seek))});
8          old_pos = parseInt(old_pos) - ((now * 30) / 2 - parseInt(old_seek));
9          old_seek = aux_seek;
10 else
11     jQuery(".anno_wrapper").css({'left' : 0});
12     jQuery("#seek_line").css({'left' : (now * 30) / 2 + "px"});
13     old_seek = (now * 30) / 2;
14     old_pos = 0;
15     if(!busy_update)
16         checkAnnotationRange(now);
17         busy_update = false;
18     old = now;

```

Listing 4.13: Timeline scroll handler

Once again, since CT has different content types with distinct properties, they also have particular display handlers. While the text, audio, mark and link are shown in the sidebar, likewise the ELAN annotations, the ink annotations have to be drawn in the canvas. Listing 4.14 illustrates the drawing function which goes through all the points in the annotation's path, calculates the correct coordinates adjusting to the current canvas and video resolution, and finally writes the lines with resource to the *lineTo* and *stroke* functions. Note that these are HTML5 *<canvas>* native functions, which allow the most varied operations in the canvas field. Whenever an ink annotation stops being fit for display, we use *clearRect* in order to clean the canvas.

Finally, the remaining features of the annotation content visualization interface, such as

square colors and annotation type filters, are created and handled through typical jQuery functions, similar to others already described in this work. It is important to note that, although the CT's annotation type filters toggle the visibility of the respective annotation container in the sidebar and the timeline, in ELAN the behaviour is slightly different since we have two levels of categories. When multiple categories are selected we handle the annotation filtering as with CT. Although, when only one category is selected, we need to hide the main level in order to show the distinct sub categories. This is done with resource to the *buildChildrenTiers* and *resetMainTiers* functions, used respectively to hide or show the main category containers.

```

1 var path = annotation.path;
2 if(path.length != 0)
3     canvas_context.moveTo(path[0].x, path[0].y);
4     canvas_context.beginPath();
5     canvas_context.strokeStyle = '#f00';
6     canvas_context.lineWidth = 1;
7     canvas_context.stroke();
8     for(var i = 1; i < path.length; i++)
9         var calc_x = x;
10        var calc_y = y;
11        if(fix_height)
12            calc_x = (x * calc_width) / nat_width;
13            calc_y = (y * fixed_video_height) / nat_height;
14        else
15            calc_x = (x * fixed_video_width) / nat_width;
16            calc_y = (y * calc_height) / nat_height;
17
18        canvas_context.lineTo(calc_x, calc_y);
19        canvas_context.stroke();

```

Listing 4.14: Canvas paint annotation

4.6 Summary

In this chapter we discuss the various technologies used, as well as the implementation decisions made throughout the system. At first we gave an overview on how Drupal's core functionalities could be used for the purposes of our system and how they were adapted to better fit the requirements of the content creation process of the archive. Then we look at the mechanisms and data requirements for the related content navigation, as well as the specific implementation details of its representation both in the archive navigation and search results. Finally, an overview of the implementation of the various multimedia content visualization, from the normal video player to the complex set of operations needed to represent and manipulate the annotated video data.

5

Evaluation

Relational navigation is not a common concept in multimedia web archiving and consequently it is natural if users do not feel comfortable using such mechanisms. The goal of this chapter is, therefore, to evaluate the archive system and the implemented solution presented in this work, through the analysis of the results of usability tests performed with various users. In order to do this, a questionnaire was created for the users to answer, providing feedback on usability and relevance of the overall system and specific functionalities.

The questionnaire consisted almost entirely of a set of tasks which the user had to perform while using the system. These tasks would then have to be rated after they have been performed, in a range of 1 to 5, corresponding to whether the user completely disagrees or agrees with it, in order to gather information on how easy they were, their relevance for the purpose of the system, among others. With this purpose, the questionnaire is divided in four distinct parts, where the first three aim to gather numeric data regarding the ease of use of some of the functionalities of the system, while the purpose of the last part is to obtain personal and general information regarding the user and the system, as well as gather important feedback through a few open answer questions regarding the system and its functionalities.

In order to follow a certain workflow while the user is answering the questionnaire, the three first sections consist of groups of related functionalities and the questionnaire itself tries to restrict the user to one group in each section, in order to keep his focus on the questions at hand. The groups are defined as *Navigation*, *Searching* and *Content Creation*, whose functionalities correspond respectively to the overall navigation of the archive, the

browsing and searching of custom content and the creation of new content in the backoffice area.

5.1 Results analysis

In this section we present part of the results obtained from the questionnaire along with its analysis, in order to gather the appropriate feedback regarding the various aspects of the system, and draw the appropriate conclusions. The analysis is divided in four different sections corresponding to the topic of the questions, either regarding the user, navigation, searching or creation of content. Both the questionnaire and its results can be found in its totality in [Appendix A](#).

5.1.1 Users

For this analysis we gathered results from 18 users where 61% of the inquired were male and the remaining 39% female, with ages ranging from 19 to 35 years old although the predominant age is 24, with 27% of the user sample. Of all the users only 2 are not familiar with current web technologies which, while not being relevant for the overall results, may have some influence in the respective user experience.

Regarding online multimedia archives, only 23% of the users do not usually make use of such platforms in order to store and share their content. This tells us that the majority of the users are familiar to this concept and should have stronger opinions regarding the functionalities tested.

5.1.2 Navigation

During this section the users were instructed to browse through the system by following some simple directions, starting from the navigation of the main page to the visualization of specific content types. The goal of the questions in this section was to gather feedback about the overall usability and interface of the archive in order to further improve it.

Regarding the main archive display, as [figure 5.1\(a\)](#) illustrates, over 70% of the users found it natural to navigate through its contents and understood the meaning of the graph and its relations, even with no previous contact with the system. When asked to find a specific content item within the main archive no user found difficulty and 78% considered it very easy, as can be seen in [figure 5.1\(b\)](#), proving that even without knowing the structure of the archive a user can easily find the intended contents.

After entering an individual archive and asked to find a specific content, it is noticeable that the users did not feel so comfortable with the relational navigation as only 6% gave

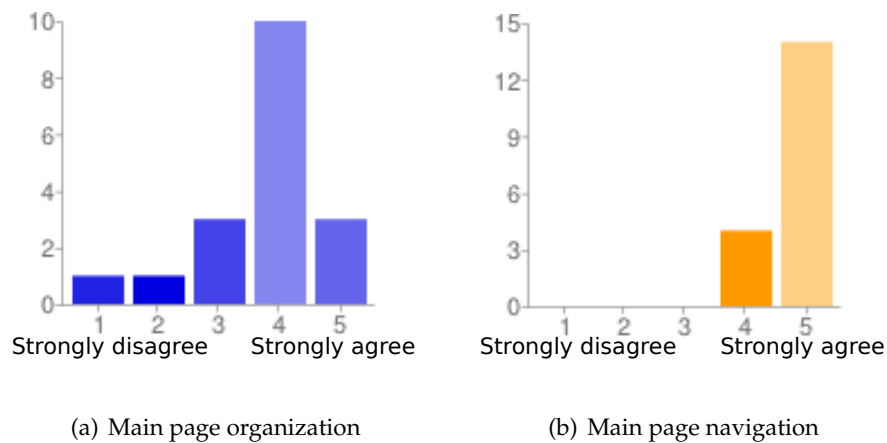


Figure 5.1: Results of the evaluation of the main archive's structure and navigation.

it the maximum rating and 28% disagreed that this functionality felt natural. This may relate to the fact that this concept is not so common in typical multimedia archives and therefore users unfamiliar with it might not know what to do when presented with the graph. This result is not alarming as more than half of the inquired found no problems with this task. However, it is good to pay attention to some interface details which could help inexperienced users.

Still in the navigation section of the questionnaire the users were told to view a Creation Tool annotated video content in order to evaluate its interface. When asked about the clarity of the annotation display throughout the video, users rated it highly as illustrated in figure 5.2(a). This result comes somehow as expected since the timeline and color code implemented for such was one of the main focus of this work. Controlling the visibility of the various annotation categories during video playback was not so natural, as almost 50% (5.2(b)) of the users did not agree that this task was easy to do.

Finalizing this part of the questionnaire the users were told to return to the previous archive navigation mode. The goal was to evaluate the functionality of the toggle button for this purpose and how natural it was to go from *Linear* to *Cloud* mode, which seemed to prove itself well, considering the fact that over 70% of the users rated it highly.

5.1.3 Searching

The main focus of this section was the archive's searching functionalities as well as the filtering options of the content in the main archive. Continuing from the individual archive, the users were indicated to proceed to use the related content area in order to evaluate

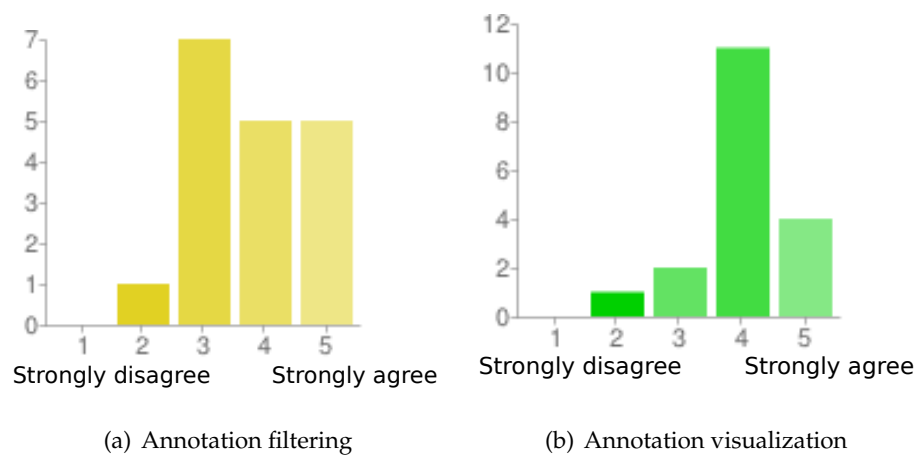


Figure 5.2: Overall annotation filtering and visualization evaluation

both the mechanism used to get there as well as the searching interface and results display. While in both cases the results were positive, where over 70% were pleased with these mechanisms, some of the negative responses, especially regarding the individual graph's context menu options, may be due to the fact that some of the interface details were not on their final version, making it less perceivable.

Regarding the content filtering of the main archive's graph, the results show that 60% of the users inquired rated this task highly in regards of its simplicity to perform. Although this value is high, the remaining percentage of users which rated it poorly helps us realize that there are still improvements to be made in this mechanism.

5.1.4 Content creation

This part of the questionnaire had focus on the content administration part of the archive. By first asking the users to create an account, which no user had any issues while doing so, they were then directed to the backoffice area of the archive, in order to create their own content and further evaluate the system regarding the corresponding mechanisms. Reaching the backoffice area through the main page represented no problem as the majority of the users considered it natural to do. The workflow for content creation also rated highly in terms of simplicity. When the users were asked to create the initial archive hierarchy 72% agreed it was rather easy to do, happening similarly for the "Slideshow" content creation (78%), where only one user did not fully understand the required steps for this task.

Finally, the users were directed to their personal archive's visualization page, in order to view how the changes made in the archive would reflect in the individual archive graph. As illustrated in figure 5.3, the majority of the users clearly perceive the changes made

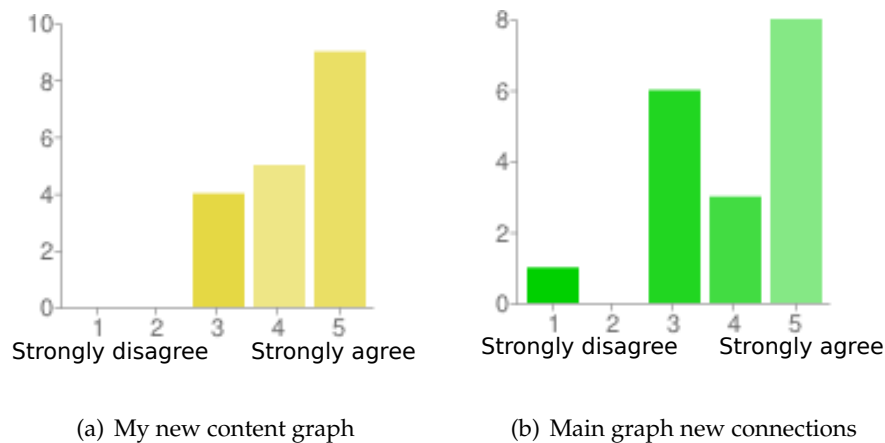


Figure 5.3: Newly created content

to both in the individual (78%) and the global archive (71%), consequence of the content created by them.

5.2 Summary

When asked about the relational navigation mechanism presented in the archive 73% of the users found it very useful and 61% did not report any difficulties while using it. From the feedback gathered through some of the open-answer items in this questionnaire, it was clear that most users felt rather lost at first due to the fact that this was the first time they were in contact with such navigation mechanism. Moreover, and mostly due to some interface issues, some users mentioned that the connections between the contents in the main graph were not clear, which makes us consider a different approach to how they are displayed. Once these connections are understood, users show an increasing positive feedback regarding them, where 89% of them consider them useful when browsing through the archive. This type of feedback was exactly what was expected for the relational navigation, where in general users consider it a rather good mechanism to research a specific context, finding content related to a specific category or person, or even when browsing without a specific purpose in which we can discover interesting content we were not expecting.

The overall results obtained are positive and indicate that, although the system may require some final functionality and interface adjustments, it already provides the users with an efficient tool for the intended archiving purposes. As illustrated in figure 5.4, the three different groups of functionalities had an average classification around 4 which tells us that the system is already at a good stage. But although the overall rating is

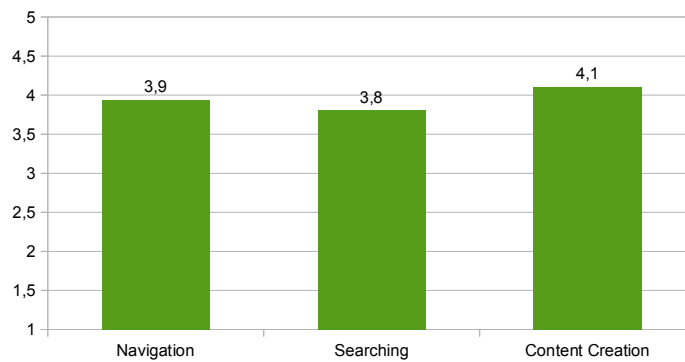


Figure 5.4: Average system ratings

high, when asked about the ease of adaptation when introduced to this system and its usability, 44% considered it challenging at part and almost 50% would not intend to use this system regularly. According to the feedback gathered from some of the users, one of the causes of this low rating was the overall design of the archive, which sometimes was not explicit enough and others too dark for users to know how to perform certain tasks. Also, the fact that the filter controls were located in the same menu blocks used for navigation would confuse users and therefore they would not naturally look for them there. Since these are all minor details which can be easily fixed and considering the fact that the archive is still under development, it was good to know that those were the main issues users encountered while testing it.



Conclusions and future work

In this thesis we presented an online platform for multimedia information archiving capable of storing and handling data from multiple users. The system should be able to create connections between content and users within the archive, as well as handling various multimedia types, including **annotated videos** originated from specific software used in contemporary dance.

The main goal of this work was therefore, to provide users the possibility to store and organize their personal multimedia content in a customizable hierarchy, creating **individual archives** within the main one, which will serve as a **knowledge base** for external use. Since this work is done in the scope of the TKB project the focus was mainly on the contemporary dance context, the creative process of choreographers and the most commonly used multimedia types by them. However, the system is implemented to be as generic and flexible as possible and therefore it is possible to easily adapt it to any area or context intended and, if necessary, extending the supported content types to fit such context. In order to meet the previous goals, it was necessary to thoroughly analyse how the connections created between content would be translated visually, resulting in the creation of the **relational navigation** and **search** featured in the archive.

The continuous process of evaluation of this work consisted in the regular analysis of the different functionalities of the system during its development. The challenges encountered along this process regarding some of the intended functionalities and the chosen technologies led to the adjustment of the interface design and, in extreme situations, to the restructure of the core of the functionality. The final step of the evaluation process was done by conducting an usability questionnaire to a set of users. These users had

no previous contact with the system and, as such, the goal was to gather the feedback required for the final adjustments of the system. The results obtained were good overall and it was also noticeable that the concept behind the relational navigation of multi-user content was easily accepted by the users and it did not pose any major challenge to the overall usability of the archive. Moreover, from the feedback gathered it was noticeable that the overall design needs some improvement as users took longer time than usual to perform easy tasks, due to the fact that they simply could not find the options intended for such.

6.1 Challenges and system limitations

The development of this system relied on the integration of different components and external libraries and, as such, the main challenges and limitations of this implementation are highly related to its dependence on:

- The core system in which it is implemented. Since the system is developed having Drupal as its core, it highly relies on it in order to maintain the designed functionalities. While the continued update of its modules may create additional features for our system, the lack of support may produce obstacles in its development;
- The implemented video playback and ink annotation visualization rely on the future of the HTML5 technology. This is already a widely supported format and might be accepted as a standard for web content, providing a wider support and compatibility with browsers and video formats. However, if such does not happen the system is somehow limited to the already existing formats, possibly forcing us to developed alternatives in order extend the support of the archive's functionalities;
- Maintaining a stable interface design throughout the increase of users and content. As this is a highly dynamic platform, where users can generate new content every day, it is necessary to devise ways of keeping the display of such items organized, in order to maintain the originally designed functionalities.

6.2 Future work

At its current state, the system developed is already a fully functional multimedia archive, as it has all the main features and requirements implemented. Nevertheless, there are some small implementation details in the existing design and functionalities which would require further adjustments. Similarly, some features initially intended for the archive and others that came during the development process were not taken further due to time restraints. As the main goal was to create a stable and usable system with the originally required functionalities could be implemented in the future of the system.

6.2.1 Content preview in graph navigation

The current relational navigation implemented does not provide much information about the specific content items besides titles and tag words, when applicable. This makes the graph navigation less natural and finding the desired contents becomes harder when the users do not know, for example, the type of content associated with each item. As such, one of the intended improvements over this functionality is to provide proper content type filtering as well as individual content preview during graph navigation. Being able to preview an item's content, even if it is static in case of videos, while navigating the video would certainly help users unfamiliar with the archive to explore and find content they would enjoy. In a more advanced version of this feature, the entire graph would consist of actual interactive pieces of content, which the user would actually be able to view in place, without having to be redirected to the individual content visualization page.

6.2.2 *ELAN* annotations searching mechanism

The annotations exported by *ELAN* have specific relevance both for the creators and people doing research on the respective content. As this software is used for extensive video analysis and, in the specific context of contemporary dance, the analysis consists of dozens of tiers annotating separate parts of a piece or rehearsal, it would be useful to develop a mechanism for efficient browsing of such text annotations. This would easily let user search for the intended keywords among the annotations, which would link directly to the respective timeslot in the video. This functionality would not be difficult to be implement. However, due to the lack of time, it was not possible to devise the appropriate interface for both the searching interface and its results within the already complex annotated video display.

6.2.3 Video transcoding

While the HTML5 specification does not broaden enough to support other browsers and video formats, there are alternatives which can be used in order to solve this issue. Since not all the users of this archive will be familiar with video formats and its conversion to the requirements of our system, video transcoding, whether done locally by the system or online through external services, would provide a solution for this. Some testing was done with a local Drupal transcoding mechanism which worked well at some extent. However, due to its limitations it was not reliable to keep it responsible for the conversion of all the videos uploaded to the system. Using an external service was considered but not tested as most of them require payment, which was not possible during the course of this work.

Bibliography

- [1] Bestiario. Accessed January 2012, <http://www.bestiario.org/>.
- [2] Dance tech. Accessed January 2012, <http://www.dance-tech.net/>.
- [3] Dancers! Accessed January 2012, <http://www.dancersproject.com>.
- [4] Data-driven documents (d3js). Accessed September 2012, <http://d3js.org/>.
- [5] Drupal vs joomla vs wordpress. Accessed January 2012, <http://foliovision.com/2011/04/02/drupal-vs-joomla-mambo-vs-wordpress>.
- [6] Elan. Accessed January 2012, <http://www.lat-mpi.eu/tools/elan/>.
- [7] Jacob's pillow dance interactive. Accessed January 2012, <http://danceinteractive.jacobspillow.org/>.
- [8] Mpeg-7 overview. Accessed January 2012, <http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm>.
- [9] The national archives experience: Digital vaults. Accessed January 2012, <http://www.digitalvaults.org/>.
- [10] Shoutex: Wordpress vs. joomla vs. drupal. Accessed September 2012, [urlhttp://shoutex.com/wordpress-vs-joomla-vs-drupal-choosing-a-cms](http://shoutex.com/wordpress-vs-joomla-vs-drupal-choosing-a-cms).
- [11] Siobhan davis replay. <http://www.siobhandaviesreplay.com/>.
- [12] Site builder shootout: Drupal vs. joomla vs. wordpress. Accessed January 2012, http://www.computerworld.com/s/article/9219685/Site_builder_shootout_Drupal_vs._Joomla_vs._WordPress.
- [13] Synchronous objects. Accessed January 2012, <http://synchronousobjects.osu.edu/>.

- [14] Tech: Open source wars: Wordpress vs drupal vs joomla. Accessed September 2012, <http://www.tech.com/2011/07/open-source-wars-wordpress-vs-drupal-vs-joomla/>.
- [15] W3schools html5 video reference. Accessed September 2012, http://www.w3schools.com/html/html5_video.asp.
- [16] Webdeveloperjuice: Comparison of wordpress drupal and joomla. Accessed September 2012, <http://www.webdeveloperjuice.com/2011/03/18/comparison-of-wordpress-drupal-and-joomla/>.
- [17] Hennie Brugman, Albert Russel, and Xd Nijmegen. Annotating multi-media / multimodal resources with elan. In *In proceedings of the International Conference on Language Resources and Evaluation (LREC, 2004)*, pages 2065–2068, 2004.
- [18] M. Butcher, J. Wilkins, and L. Garfield. *Drupal 7 Module Development*. Community experience distilled. Packt Publishing, 2010.
- [19] Diogo Cabral, João Valente, João Silva, Urândia Aragão, Carla Fernandes, and Nuno Correia. A creation-tool for contemporary dance using multimodal video annotation. In *Proceedings of the 19th Association for Computing Machinery (ACM) international conference on Multimedia*, MM '11, pages 905–908, New York, NY, USA, 2011. ACM.
- [20] Miguel Costa, Nuno Correia, and Nuno Guimarães. Annotations as multiple perspectives of video content. In *Proceedings of the tenth Association for Computing Machinery (ACM) international conference on Multimedia*, MULTIMEDIA '02, pages 283–286, New York, NY, USA, 2002. ACM.
- [21] van Riel C. Eliens A. and Wang Y. Navigating media-rich information spaces using concept graphs - the abramovic dossier. In *Proc. International Conference on Multidisciplinary Information Sciences and Technologies (InSciT2006)*, pages 25–28, 2006.
- [22] Joey Hagedorn, Joshua Hailpern, and Karrie G. Karahalios. Vcode and vdata: Illustrating a new framework for supporting the video annotation workflow. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, pages 317–321, New York, NY, USA, 2008. ACM.
- [23] I. Herman, G. Melancon, and M.S. Marshall. Graph visualization and navigation in information visualization: A survey. *Visualization and Computer Graphics, IEEE Transactions on*, 6(1):24–43, jan-mar 2000.
- [24] David Hyatt and Ian Hickson. HTML 5. W3C working draft, W3C, August 2009. <http://www.w3.org/TR/2009/WD-html5-20090825/>.

- [25] Michael Kipp. Anvil - a generic annotation tool for multimodal dialogue. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 1367–1370, 2001.
- [26] Wei Lai, Xiaodi Huang, Quang Vinh Nguyen, and Mao Lin Huang. Applying graph layout techniques to web information visualization and navigation. In *Computer Graphics, Imaging and Visualisation, 2007. CGIV '07*, pages 447–453, aug. 2007.
- [27] Stefanie Müller, Gregor Miller, and Sidney Fels. Using temporal video annotation as a navigational aid for video browsing. In *Adjunct proceedings of the 23rd annual ACM symposium on User interface software and technology, UIST '10*, pages 445–446, New York, NY, USA, 2010. ACM.
- [28] A. Russel P. Berck. Annex- a web-based framework for exploiting annotated media resources. In *Proceedings of the International Language Resources and Evaluation (LREC, 2006)*.
- [29] Vikash Singh, Celine Latulipe, Erin Carroll, and Danielle Lottridge. The choreographer’s notebook: a video annotation system for dancers and choreographers. In *Proceedings of the 8th ACM conference on Creativity & Cognition (C&C)*, pages 197–206, New York, NY, USA, 2011. ACM.
- [30] Andre Suslik Spritzer and Carla M. D. S. Freitas. Navigation, interaction in graph visualizations. In *Revista de Informatica Teorica e Aplicada - RITA*, 2008.
- [31] Joao Gaspar Valente. Sistema multimodal para captura e anotacao de video. Master’s thesis, Faculdade de Ciencias e Tecnologia da Universidade Nova de Lisboa, FCT - UNL, 2011.
- [32] J. VanDyk and T. Tomlinson. *Pro Drupal 7 Development*. Apresspod Series. Apress, 2010.
- [33] Peter Wittenburg, Hennie Brugman, Albert Russel, Alex Klassmann, and Han Sloetjes. Elan: a professional framework for multimodality research. In *In Proceedings of the International Language Resources and Evaluation Conference (LREC, 2006)*.



Appendix: Questionnaire

A.1 Questions

Navigation

In the main page of the archive browse through the content, getting familiar with the interface and the filtering mechanisms particularly.

The goal is for you to follow the tasks listed and answer whether you agree with the statement, in a scale of 1 to 5.

1. The content organization of the main page is well organized and I was able to understand how to navigate through it.

12345

Strongly disagree / Strongly agree

2. I easily found the name "Rui Horta" in order to enter his personal archive.

12345

Strongly disagree / Strongly agree

3. It felt natural to browse through the archive, finding the "Creation Tool Video Example" content item and proceed to its visualization.

12345

Strongly disagree / Strongly agree

4. After starting the video playback, I easily filtered the different annotation types and

toggled the visibility of the sidebar and timeline.

12345

Strongly disagree / Strongly agree

5. The annotations displayed throughout the video playback were clear and I easily associated them with its position in the timeline.

12345

Strongly disagree / Strongly agree

6. I easily understood how to return to previous archive navigation, or "Cloud mode".

12345

Strongly disagree / Strongly agree

Searching

The following questions regard overall searching in the archive as well as specific content related data and filtering capabilities.

7. I easily found related content with the options displayed while navigating through content in the archive.

12345

Strongly disagree / Strongly agree

8. The related content results are explicit and the relations between them are easily perceivable.

12345

Strongly disagree / Strongly agree

9. Searching the contents I wanted was easy and the results were well organized.

Try different combinations of search inputs and result display options in order to get familiarized with the system, then go back to the home page.

12345

Strongly disagree / Strongly agree

10. Back in the main page I felt comfortable filtering the available content items in order to see only those related to "dance", and it was easy to find the "Labs Gallery".

Browse through the gallery and other content for a while, then return to the main page.

12345

Strongly disagree / Strongly agree

11. textbfCreating content

In this part we access content creation and therefore a user account is necessary.

12. As an anonymous user it was easy to register a new account and the process was

simple.

Register a user account with the mandatory fields in order to perform the following tasks

12345

Strongly disagree / Strongly agree

13. Already registered and logged in, it was natural to go from the main page of the archive to the backoffice area in order to create new content.

12345

Strongly disagree / Strongly agree

14. The content creation workflow felt natural and it was easy to create the desired hierarchy for my archive.

Create at least one "Menu" and "SubMenu" items and then proceed to create a new "Content" item.

12345

Strongly disagree / Strongly agree

15. While creating a new image "Slideshow" content item the whole process was natural, I understood what to do in each step without problems.

Be sure to add a few tags to the new content.

12345

Strongly disagree / Strongly agree

16. When trying to create a new "Content" item, it was easy for me to understand the different options available, their differences and purposes.

12345

Strongly disagree / Strongly agree

17. After finishing the content creation, I was able to instantly verify the changes in "My Content" and the created hierarchy reflected in the navigation.

12345

Strongly disagree / Strongly agree

18. When returned to the main page of the archive I was able to observe the relations between the newly created content and the pre-existing ones.

12345

Strongly disagree / Strongly agree

General

19. Are you familiar with current web technologies.

Yes / No

20. I regularly use online multimedia archives to store and share my contents.

12345

Strongly disagree / Strongly agree

21. The type of navigation and organization of content in the archive is useful for this type of system.

12345

Strongly disagree / Strongly agree

22. Did you have any difficulties adapting to this type of content navigation?

Yes / No

If you answered "Yes", please explain briefly which ones.

23. Do you think the connections between different content items are useful when browsing through content or exploring specific categories?

Yes / No

Please explain briefly why.

24. Overall I found it easy to use the system.

12345

Strongly disagree / Strongly agree

25. I would use this system to store my multimedia contentwork.

12345

Strongly disagree / Strongly agree

26. Would you change anything in this system?

27. Gender

Male / Female

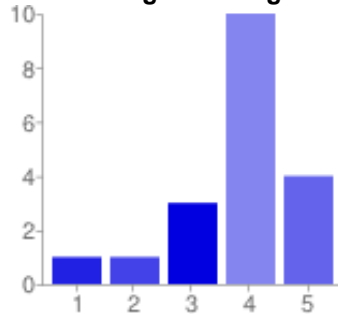
28. Age

A.2 Results

Navigation

In the main page of the archive browse through the content, getting familiar with the interface and the filtering mechanisms particularly. The goal is for you to follow the tasks listed and answer whether you agree with the statement, in a scale of 1 to 5.

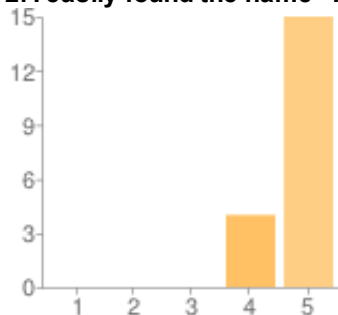
1. The content organization of the main page is well organized and I was able to understand how to navigate through it.



Strongly disagree Strongly agree

1 -Strongly disagree	1	5%
2	1	5%
3	3	16%
4	10	53%
5 -Strongly agree	4	21%

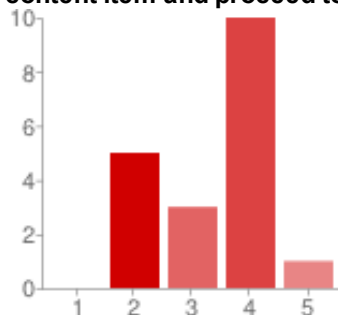
2. I easily found the name "Rui Horta" in order to enter his personal archive



Strongly disagree Strongly agree

1 -Strongly disagree	0	0%
2	0	0%
3	0	0%
4	4	21%
5 -Strongly agree	15	79%

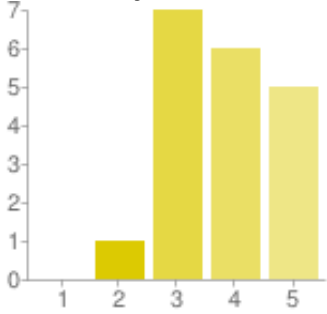
3. It felt natural to browse through the archive, finding the "Creation Tool Video Example" content item and proceed to its visualization



Strongly disagree Strongly agree

1 -Strongly disagree	0	0%
2	5	26%
3	3	16%
4	10	53%
5 -Strongly agree	1	5%

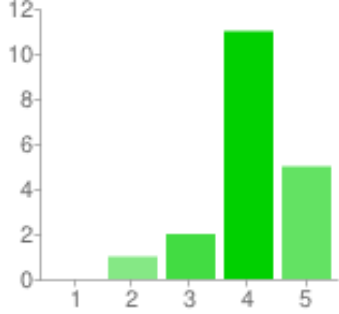
4. After starting the video playback, I easily filtered the different annotation types and toggled the visibility of the sidebar and timeline.



Strongly disagreeStrongly agree

1 -Strongly disagree	0	0%
2	1	5%
3	7	37%
4	6	32%
5 -Strongly agree	5	26%

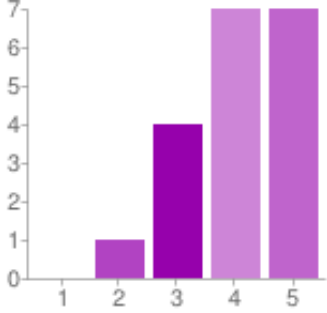
5. The annotations displayed throughout the video playback were clear and I easily associated them with its position in the timeline.



Strongly disagreeStrongly agree

1 -Strongly disagree	0	0%
2	1	5%
3	2	11%
4	11	58%
5 -Strongly agree	5	26%

6. I easily understood how to return to previous archive navigation, or "Cloud mode".



Strongly disagreeStrongly agree

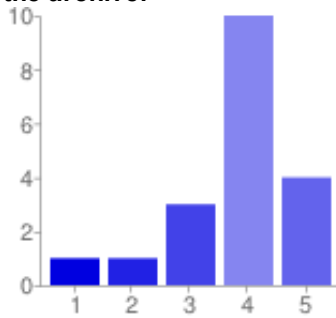
1 -Strongly disagree	0	0%
2	1	5%
3	4	21%
4	7	37%
5 -Strongly agree	7	37%

Searching

The following questions regard overall searching in the archive as well as specific content related data and filtering capabilities.

7. I easily found related content with the options displayed while navigating through content in

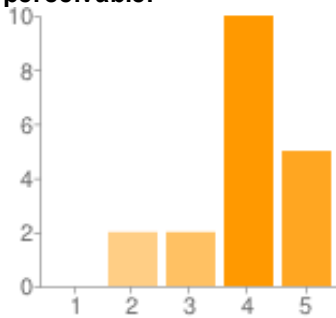
the archive.



Strongly disagreeStrongly agree

1 -Strongly disagree	1	5%
2	1	5%
3	3	16%
4	10	53%
5 -Strongly agree	4	21%

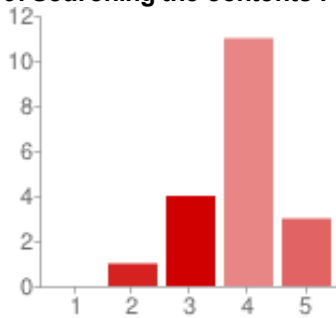
8. The related content results are explicit and the relations between them are easily perceivable.



Strongly disagreeStrongly agree

1 -Strongly disagree	0	0%
2	2	11%
3	2	11%
4	10	53%
5 -Strongly agree	5	26%

9. Searching the contents I wanted was easy and the results were well organized.

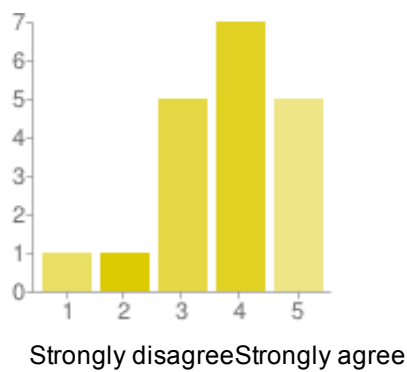


Strongly disagreeStrongly agree

1 -Strongly disagree	0	0%
2	1	5%
3	4	21%
4	11	58%
5 -Strongly agree	3	16%

10. Back in the main page I felt comfortable filtering the available content items in order to see only those related to "dance", and it was easy to find the "Labs Gallery" .

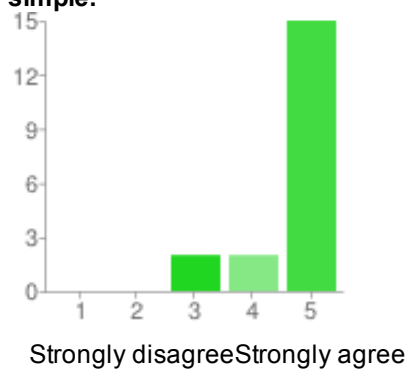
1 -Strongly disagree	1	5%
2	1	5%
3	5	26%
4	7	37%
5 -Strongly agree	5	26%



11. Creating content

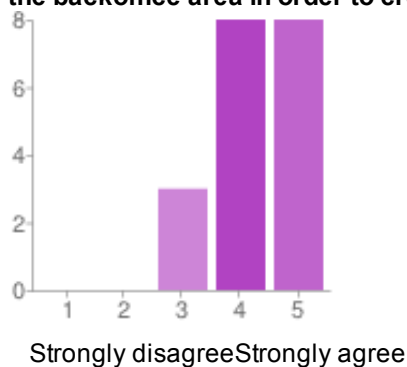
In this part we access content creation and therefore a user account is necessary.

12. As an anonymous user it was easy to register a new account and the process was simple.



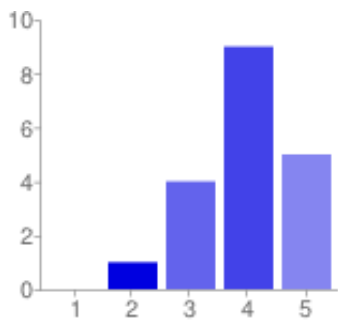
1 -Strongly disagree	0	0%
2	0	0%
3	2	11%
4	2	11%
5 -Strongly agree	15	79%

13. Already registered and logged in, it was natural to go from the main page of the archive to the backoffice area in order to create new content.



1 -Strongly disagree	0	0%
2	0	0%
3	3	16%
4	8	42%
5 -Strongly agree	8	42%

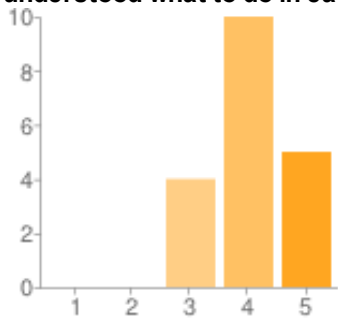
14. The content creation workflow felt natural and it was easy to create the desired hierarchy for my archive.



Strongly disagree Strongly agree

1 -Strongly disagree	0	0%
2	1	5%
3	4	21%
4	9	47%
5 -Strongly agree	5	26%

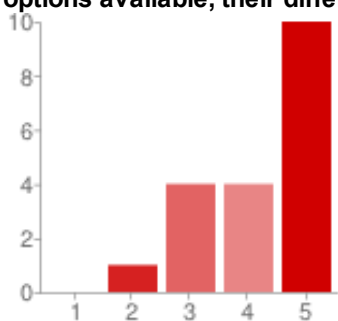
15. While creating a new image "Slideshow" content item the whole process was natural, I understood what to do in each step without problems.



Strongly disagree Strongly agree

1 -Strongly disagree	0	0%
2	0	0%
3	4	21%
4	10	53%
5 -Strongly agree	5	26%

16. When trying to create a new "Content" item, it was easy for me to understand the different options available, their differences and purposes.

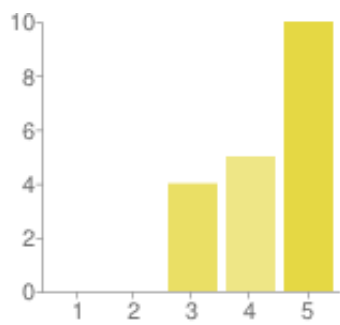


Strongly disagree Strongly agree

1 -Strongly disagree	0	0%
2	1	5%
3	4	21%
4	4	21%
5 -Strongly agree	10	53%

17. After finishing the content creation, I was able to instantly verify the changes in "My Content" and the created hierarchy reflected in the navigation.

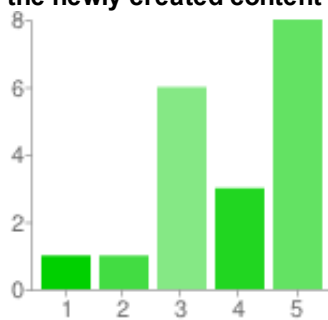
1 -Strongly disagree	0	0%
2	0	0%
3	4	21%
4	5	26%



5 -Strongly agree **10** 53%

Strongly disagreeStrongly agree

18. When returned to the main page of the archive I was able to observe the relations between the newly created content and the pre-existing ones.

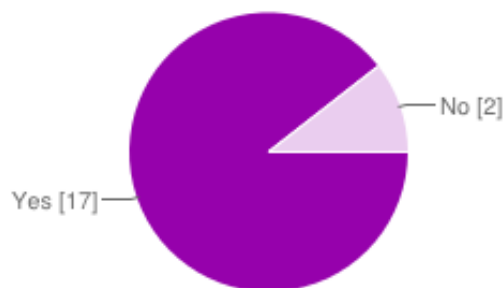


1 -Strongly disagree	1	5%
2	1	5%
3	6	32%
4	3	16%
5 -Strongly agree	8	42%

Strongly disagreeStrongly agree

General

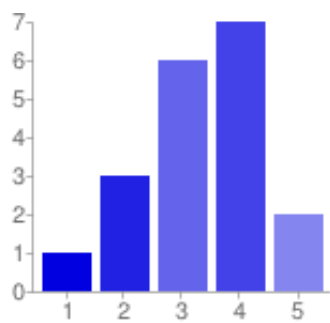
19. Are you familiar with current web technologies.



Yes	17	89%
No	2	11%

20. I regularly use online multimedia archives to store and share my contents.

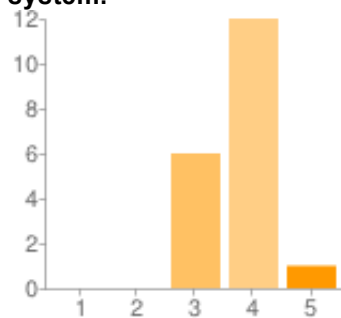
1 -Strongly disagree	1	5%
2	3	16%



Strongly disagree Strongly agree

3	6	32%
4	7	37%
5 -Strongly agree	2	11%

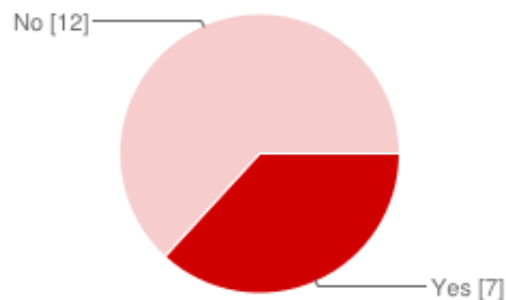
21. The type of navigation and organization of content in the archive is useful for this type of system.



Strongly disagree Strongly agree

1 -Strongly disagree	0	0%
2	0	0%
3	6	32%
4	12	63%
5 -Strongly agree	1	5%

22. Did you have any difficulties adapting to this type of content navigation?

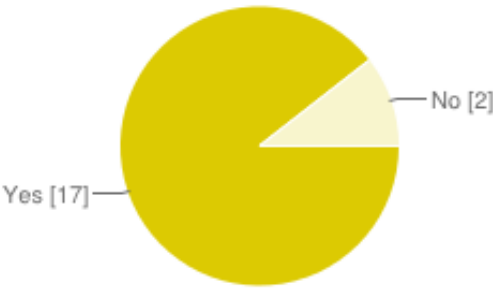


Yes	7	37%
No	12	63%

If you answered "Yes", please explain briefly which ones.

Porque nunca tinha trabalho com este sistema. Tive de me habituar à navegação nos menus e nos conteúdos. The view switching approach could be more clear. I believe the cloud/linear mode button should have a bigger focus on the main screen. The same button also appeared to be kind of "buggy" because there were time where I tried to switch to linear mode and the screen switched to cloud mode, and vice-versa. Content creation could be a little bit more intuitive. It's not intuitive the fact that the 'breadcrumbs' along the navigation page are also filters. Once you've figured it out its apparent ...

23. Do you think the connections between different content items are useful when browsing through content or exploring specific categories?

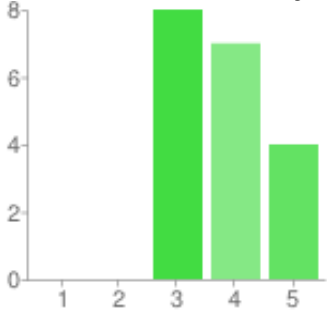


Yes	17	89%
No	2	11%

Please explain briefly why.

Por se tratar de uma forma de organização intuitiva que facilita a pesquisa por uma área específica, por exemplo. Dá para perceber as ligações entre várias pessoas e ver de que forma o conteúdo é semelhante ou não. I can't remember any special example where that would be useful for my personal purposes. Because its easier to find content if we have a relationship between content and tags. Permite uma melhor navegabilidade entre os diversos conteúdos, de uma forma intuitiva e rápida. Torna-se mais fácil explorar conteúdos específicos Yes, because it permits the user to discover new items, that may ...

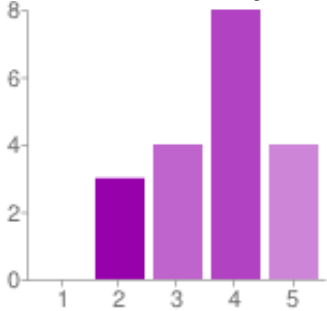
24. Overall I found it easy to use the system.



1 -Strongly disagree	0	0%
2	0	0%
3	8	42%
4	7	37%
5 -Strongly agree	4	21%

Strongly disagreeStrongly agree

25. I would use this system to store my multimedia content/work.



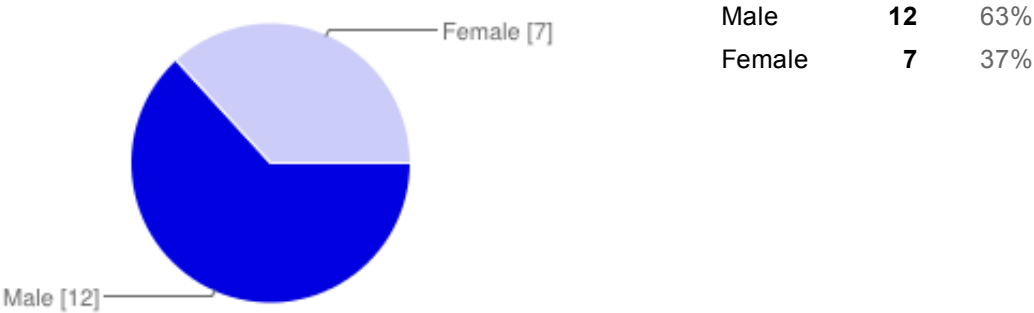
1 -Strongly disagree	0	0%
2	3	16%
3	4	21%
4	8	42%
5 -Strongly agree	4	21%

Strongly disagreeStrongly agree

26. Would you change anything in this system?

O esquema de cores pois considero o actual um pouco escuro. Outras questões de design como as as caixas associadas a cada um dos itens ramificados. Colocar algumas opções mais visíveis. Facilitar o juntar as tags na parte de criar conteúdo. Besides correcting some bugs, I would use more colors to evidence some resources such as cloud/linear button. The dark theme is too limited. The tree hierarchy should appear above the interaction screen, instead of below. I had an error when i was registering. It said that my email was wrong and also said that everything was alright. que me lembre nada De mome ...

27. Gender



28. Age

23 24 24 24 25 22 24 25 22 23 22 23 25 23 24 24 25 24 25

Number of daily responses

