NTNU
Norwegian University of
Science and Technology

Exploring the navigational faceted search

Mei Jain Fung

Master of Science in Computer Science
Submission date: February 2018
Supervisor: Trond Aalberg, IDI

Norwegian University of Science and Technology
Department of Computer Science

Summary

Today we have access to a broad range of information right at our fingertips. For decades, the technology for information retrieval has assisted humans in information seeking. While the conventional search approaches work well in retrieving information when the user knows what information to look for, user studies show that the user has a harder time finding relevant information when the user wants to discover new information about a specific topic.

Much work has been done in the library domain in the attempt to describe features and relationships between different bibliographic entities, such as in the FRBR model. However, usage of these models in real-world applications has been less systematically explored and users still experience difficulties scanning through a long list of information. Luckily in recent years, faceted search has become increasingly common in online information access, including bibliographic catalogues, and more effort is being put into the design of the user interface, addressing the weaknesses in conventional search approaches.

The thesis starts with giving the reader an introduction to theory around faceted search and look at some successful design patterns for search applications. Further, the implementation made and the functionality will be explained on the search application BIBSURF, with an information collection organized by the FRBR model. Finally, the researcher examines how the different filtering strategies perform, both by analyzing use cases and perform user studies. To conclude, the ultimate goal of this thesis is to provide future practitioners on this topic a contribution of a usability study to explore the quality of the user interface of search applications supporting non-professional searchers in rich information seeking tasks.

Preface

My interest in web development, especially development on the client side, lead me to choose this master project. This project has given me the opportunity to aid in designing a user interface for a search application and to learn React js, a popular yet powerful JavaScript library for building user interfaces.

Only in the last decade, the design of user interfaces has been user-centered focusing more on the usability. User studies show that search applications incorporating navigation have high success and more search applications are adapting to this trend, particularly online bibliographic catalogues, and e-commerce. However, much more research and user studies need to be done to be able to create powerful user interfaces to fully exploit the rich and complex information collection available today.

Furthermore, writing best practice programs are talked and discussed a lot and the knowledge is usually gained through years and years of writing programs. However, few best practices are documented in detail. I want to use this opportunity to additionally document my findings on best practices for developing web applications.

Starting with only basic knowledge of user interface design, HTML, CSS and JavaScript, I am happy to have learned to adapt to learning new skills and technologies. I would like to especially thank my supervisor, Trond Aalberg, for all the helpful advice and support throughout the past two semesters. Further, I would like to thank my family and friends for the constant support and thank all the people participating in this user study!

Table of Contents

List of Tables

List of Figures

Abbreviations

BIBSURF	Discover Bibliographic Entities by Searching for Units of Interest, Ranking and Filtering
CRA	Create React App
CSS	Cascading Style Sheets
DOM	Document Object Model
Flamenco	FLexible information Access using MEtadata in Novel COmbinations
FR	Functional Requirement
FRBR	Functional Requirements for Bibliographic Records
HTML	Hypertext Markup Language
HTTP	HyperText Transport Protocol
HTTPS	Hypertext Transfer Protocol Secure
IFLA	International Federation of Library Associations and Institutions
IR	Information Retrieval
JS	JavaScript
JSX	JavaScript XML
JSON	JavaScript Object Notation
NTNU	Norges teknisk-naturvitenskapelige universitet (Norwegian University of Science and Technology)
Q	Question
RDF	Resource Description Framework
RF	Relevance Feedback
RQ	Research Question
UI	User interface
UML	Unified Modeling Language
uuid	Universally Unique Identifier
url	Uniform Resource Locator
UX	User Experience
WWW/ web	World Wide Web
XML	Extensible Markup Language

Chapter 1

Introduction

As digital information is increasing on the World Wide Web (WWW/ Web), search systems need to support more effective searches in aiding end users to rapidly find the information they are looking for. Search applications from different domains are now focusing more on the user experience (UX) and design. Today, users expect search interfaces to provide some degree of filtering on the collection of items. Just to give some examples, the domain of e-commerce, filtering mechanisms have become de facto standard providing filters for price, product description, rating, and brand. Digital libraries have also benefited from the rich data structure that is available in the information collection, providing filters for publication date, type of media and roles like author and translators. Lastly, the area of video creation has exploded on the web in the last decade. Looking at the video-sharing website, YouTube[1], statistics from 2016 show that 300 hours of video are uploaded to YouTube every minute and 4,950,000,000 videos are viewed on YouTube everyday (StatisticBrain, 2016). Search application for video collections can, for example, provide filters for genre, upload date, and video duration, from companies like Netflix[2], Amazon[3] and YouTube.

One objective of information retrieval systems is to help the user understand the options and content of the database the user is searching. However, (Ercegovac, 2006) sites that user studies point out that end users face problems when searching in online library catalogs particularly when scanning through long displays. Fortunately, usability study shows that search interfaces supporting both keyword search and browsing of the retrieved result set have high success for search in relatively large information collections [(English et al., 2002)]. In addition, utilizing the consistency in design across different websites minimizes the learning time, and users immediately know how to interact with the web page. Since consistency in UI design increases the familiarity and predictability, which increases the usability.

[1]https://www.youtube.com/
[2]https://www.netflix.com/
[3]https://www.primevideo.com/

The International Federation of Library Associations and Institutions[4] (IFLA), is an international organization, promoting a high standard of provision and delivery of library and information services. The Functional Requirements for Bibliographic Records (FRBR) is a conceptual, entity-relationship model developed by the IFLA Study Group as a generalized view of the bibliographic universe (IFLA, 1998). FRBR provides a new set of concepts with an update of terminologies and relationships around bibliographic entities as an opportunity to re-examine existing cataloging rules and principles. IFLA together with other interested parties encourages the use and evolution of the FRBR model in bibliographic search applications to facilitate international standardization and to reduce costs for cataloguing on a global scale (Tillett, 2003).

Improving search interface, particularly bibliographic catalogues, would be useful such as in technical, environmental, earth sciences, pharmaceutical and biomedical literature where resources are likely to evolve over a longer period of time within large projects, where multidisciplinary teams working together, and among distributed laboratories. In addition, to aid people, who do not specialize in search or people with only basic knowledge of how to use computers, in using digital search systems.

This master thesis is a continuation from the previous course, TDT4501 Computer Science, Specialization Project, Spring 2017 (Fung, 2017) where the researcher implemented the client side using React on the search application BIBSURF, a search system for ranking and filtering of bibliographic RDF data, developed by researchers at NTNU, Trondheim, and University of Ljubljana, Slovenia (Aalberg et al., 2016). The master thesis continues with a focus on the usability of filters in a search application with data following the FRBR model. The researcher has in addition implemented the support for different filtering strategies for conjunction ("and"), disjunction ("or"), a combination of "and" and "or" and different views for displaying the information collection formed with the FRBR model.

1.1 Project Description

To summarize, the objective of this project is:

- implement the client side of BIBSURF, using the React js library;
- design and implement strategies for refinement of search results with "and", "or" and "andor";
- develop an approach to view the entire hierarchy tree of FRBR entities or only branches of the tree;
- and conduct a user study to explore the interpretation of the filtering mechanism with a novel design approach from the perspective of end users.

Table 1.1, specifies the research questions for this thesis.

[4] `https://www.ifla.org/`

Id	Description
RQ-1	How can we implement a dynamic user interface for search application with data formed with the FRBR model?
RQ-2	How can we best present categories to the end user in a neat and intuitively way?
RQ-2.1	Does the user understand the design associated with the categories? (Categories are attached with two numbers, the former representing the total number of results matching the categories and the latter with the total number of available results that matches the categories.)
RQ-3	Does the user understand the refined result when categories are applied to the search?

Table 1.1: Research Questions

1.2 Research Methodology

The project is divided into three main parts: a literature study, an implementation part, and a user study.

The literature study started with exploring relevant theory and research studies around the topic of this master thesis. The bibliography of books, conference documents and journal articles from the master thesis in Spring 2017 by Dana Music, a study analyzing strategies and techniques for filtering of search results (Music, 2017), was used as a starting point of the literature study.

The implementation followed the guidelines for developing a React application: breaking the UI into a component hierarchy to create simple, reusable and testable views. In addition, from the literature study, a set of design approaches was implemented following a set of design principles.

Lastly, the user study followed the user study conducted Spring 2017 (Aalberg et al., 2017) and the user tests were analyzed with a combination of observations during tests, and users' reflections from surveys and informal interviews after the tests.

1.3 Thesis Outline

Chapter 2: Background presents relevant theory around faced search.

Chapter 3: Implementation and design explains the structure of BIBSURF's information collection, the system's functionality, and implementation.

Chapter 4: Experiment describes in detail the test plan used for conducting a user study on the implemented system.

Chapter 5: Discussion presents the end results of the user study, including conclusions.

Chapter 2

Background

This chapter aims to present relevant theory around the faceted search for supporting open-ended and knowledge discovery tasks. Section 2.1 presents a general grouping of user tasks when pursuing answers using search interfaces and section 2.2 continues in explaining what is meant by exploratory search. Section 2.3 and 2.4 explains the key concepts of faceted classification and faceted navigation before diving into the topic of faceted search in section 2.5. Finally, section 2.6 wraps the chapter by presenting some design principles for building effective search interfaces.

2.1 The user task

Information can be found everywhere and the demand for robust search applications are growing from researchers and practitioners. What is a good user interface (UI) for search, depends highly on the type of answers that the users are pursuing (Hearst et al., 2002). Generally, user tasks can be divided into three categories:

1. **Fact-finding**: e.g. What is the capital of China?
2. **Open-ended**: e.g. Find a good crime novel to read.
3. **Text mining**[1]**/ knowledge discovery**: e.g. What are some promising untried treatments for Alzheimers Disease?

Tasks like finding the capital of a country, only requests for short and precise answers. Standard search systems let the user do simple lookups with keywords (e.g. ”capital China”) or directly with a natural language question (e.g. ”What is the capital of China?”).

[1]`https://www.ibm.com/support/knowledgecenter/en/SS3RA7_17.1.0/ta_guide_ddita/textmining/shared_entities/tm_intro_tm_defined.html` accessed: 6. February 2018. Text mining is the process of analyzing collections of textual materials in order to capture key concepts and themes and uncover hidden relationships and trends without requiring that you know the precise words or terms that authors have used to express those concepts.

Open-ended tasks like finding a book to read typically require some browsing in the available collection when the user does not have any specific in mind. Considering the mentioned task of finding a book to read, a user normally wants to look through books in a specific genre, from a specific author, or books with the highest review ratings. Further, the user may change the task criteria during the search process, for example, to start looking for popular books or books in a different genre if the user notices interesting books from a well-known author or gets inspired by positive or negative book reviews from other users. Search systems should, therefore, support browsing on a collection with specific attributes, and support a dynamic exploration flow without any interruptions. Lastly, when discovering answers to questions like finding treatments for diseases, it typically involves the ability of tracking trails of reasoning, performing comparisons, summarizing, and processing the information in detail.

The information retrieval (IR) research community has for decades aided with techniques and methods for manipulating, storing, retrieving and distributing information. Traditional IR systems usually provide a textual query or direct search and when queries are carefully specified, the systems returns back a ranked list of precise results. This model has for a long time dominated database management systems and is used in major search engines provided by companies like Google[2], Yahoo![3] and Microsoft[4]. These systems work well for tasks like fact-finding or question-answering scenarios where the user usually have a clue that the information exists (White and Roth, 2009).

However, traditional IR systems are not sufficient for finding answers to open-ended or knowledge discovery tasks. Luckily, more and more systems are adapting to interfaces that support what is called ”exploratory search”. Both open-ended and knowledge discovery tasks share the search and the browsing component that are required to support exploratory search. However, an analysis of search systems fully capable of supporting knowledge discovery tasks is out of scope for this master thesis. This chapter will, therefore, focus on search systems aiding users in answering open-ended tasks, where users execute what is called ”exploratory search”.

2.2 Exploratory search

With the vast amount of information highly accessible to users, search applications should aid users in solving complex problems where the user is uncertain of whether the information being sought exists. Exploratory search can be defined as information-seeking problems that are open-ended, persistent and multifaceted, where the process typically is opportunistic, with multiple iterations and by using multiple tactics before achieving a final goal (White and Roth, 2009). An exploratory search may require days, weeks or even months, where users repeat the process of navigating through relevant documents in a collection, exploiting the information, and making decisions about their next steps. Ex-

[2]https://www.google.com
[3]https://www.yahoo.com
[4]https://www.bing.com

ploratory search can, therefore, have a significant impact on a user's personal development and requires capable search applications that support users needs.

With the growth of digital information on the web, search applications face several challenges. The following list is some challenges that are often mentioned in literature:

- **Zero-hit**: too specific or ill-formed search queries entered by the user can lead the system to return an empty result set. Shneiderman states that up to 30% of searches at some services have zero-hit outcomes (Shneiderman et al., 1997).
- **Information overload**: on the contrary if a vague query is performed or searches on an information rich-topic, the system can return back a huge number of hits, leading to distractions where users overlook the relevant information displayed among the result set.
- **Difficulty forming queries with Boolean expressions**: online bibliographic systems today support Boolean queries, an example is NTNU's university library[5] in the advanced search, users can form queries with the Boolean operators for ”and”, ”or” and ”not”. Unfortunately, (Hearst, 1999) sites that user studies show that users have difficulties in forming Boolean queries and often misjudges the retrieved results. Users with limited knowledge of Boolean syntax may be confused by the semantics of the natural language. As an illustration, a query for ”chicken and pig” would indicate documents for topics for chicken and pig rather than documents for both the topics at the same time, while a query for ”chicken or pig” would indicate documents where the topics are separated.
- **Difficulty using correct terminology**: search queries are build from users existing knowledge, and users may not recall or know the specific terminologies used in literature. Besides, manual query formulation or reformulation process is an intellectual task where end users find it difficult to formulate queries relevant to their information needs.
- **Diverse user tasks**: as the definition of exploratory search states, users usually are uncertain of what the end goal might be or what information is needed. Systems should, therefore, support exploratory search at the same time support rapid lookups when users know what to look for.
- **User feeling lost or overwhelmed**: exposing the user to excessive or disorganized information may distract the user from overlooking the relevant information and cause confusion about what the user is suppose to do next.

In the process of exploratory search, search applications should aid the end users with formulating their queries and clarifying any vague information needs, learn from the existing information collection, investigate solutions to the information problems and help keep track of the progress of their search. To guide the development of future search applications in solving the mentioned challenges, (White and Roth, 2009) have collected a list of features that must present to support exploratory search activities:

[5]https://www.ntnu.no/ub

1. **Support querying and rapid query refinement**
   As users typically have difficulties forming their own search queries, systems should therefore help users formulating and refining queries in real-time. Some common techniques in the IR community includes relevance feedback (RB), query suggestion and query completion. In relevance feedback, the user has the possibility to give implicit or explicit feedback on the relevance of the information retrieved in user's initial search. The feedback is then further used by the system to retrieve relevant queries or documents to the user based on the previous information that was relevant to the user (Salton and Buckley, 1990). Query suggestion gives suggestions of queries to use after user's initial search, based on the search history of other users. And query completion provides query suggestions, most commonly appearing as a drop-down below the search box, when the user starts typing, to support the user in building their query.

2. **Offer facets and metadata-based result filtering**
   Users usually express a desire for search interfaces to organize the search results into meaningful groups, to help the users get an overview of the results and to help with deciding what to do next. User study also show that users even changes their search habits when search interfaces provides structural groupings of the result set (Kules and Shneiderman, 2008). It is possible to group a collection of information using different methods, and many open research questions discusses how to generate useful groupings. Two popular methods are clustering and faceted categorization. Clustering groups items according to some measure of similarity (Hearst, 2006). For example in document clustering, similarity is computed by common words and phrases. Clustering is an automated process and can easily be applied to text collections. However, some weaknesses of clustering includes the lack of predictability and difficulty combining multiple dimensions in a meaningful manner or labeling the groupings. Faceted categorization defines a set of meaningful categories hierarchies describing specific characteristic that are assigned to items. A collection can therefore be easily grouped together across multiple dimensions. The biggest drawback with faceted categorizations, is that the category hierarchies is build by hand and assignment of categories to items can only be automated with a certain accuracy. Faceted categorization will be discussed more later in this chapter.

3. **Leverage search context**
   As mentioned, the user queries are not well defined according to their information needs. Systems should therefore collect as much information about the user and the task context, either asking the user explicit to rank useful queries or implicitly monitoring user's interaction behavior, to resolve any query ambiguity (e.g. a query for ”Apple” may mean the technology company or simply a fruit).

4. **Offer visualizations to support insight and decision making**
   Humans are more aware of images and visual information (Hearst, 1999). For example, consider the contrast of describing the feature of a human face versus an image of the person or analyzing a table of numbers versus the same information displayed as a scatter plot. Graphical visualization of the result set can therefore help users

understand and analyze the available data and to support hypothesis generation and trend spotting.

5. **Support learning and understanding**
   Systems should present information to the user depending on user's current knowledge and skill level to support learning and understanding (two important aspects of exploratory search) to increase users confidence and feeling in control during problem solving.

6. **Facilitate collaboration**
   Different people have different perspectives, experiences, expertise, and vocabulary. And collaboration between multiple users to solve a common problem task can be more effective than searching information individually. Systems should allow for multiple users to compose queries or examine search results. In e-commerce, recommendation methods are used a lot to recommend items that other users also have purchased and liked.

7. **Offer histories, workspaces, and progress updates**
   Information searching is an iterative process where it typically involves examining multiple information sources and perform multiple search sessions that users have to keep track on. Search systems should therefore offer structured user interaction records, ”workspaces” and progression status, to support the users in revisiting results efficiently, to help organize their work progress, and keep track on where the users are located in the search process. One example of a history mechanism relevant for this study, is the use of ”breadcrumbs”. Breadcrumbs are usually displayed as a button or a link reflecting the sequence of links or searches the user have clicked or searched on since the beginning of the search process. From its functionality, the name has been named after Grimm's fairy tale of Hansel and Gretel where Hansel leaves a trail of breadcrumbs to remember their path. (Nielsen, 2007) states that user testing show many benefits and no downsides to breadcrumbs.

8. **Support task management**
   Allowing for persistent storage of session state, including queries used, document viewed and marked and paths followed, to be retrieved in later sessions, help users to investigate multiple sessions simultaneously and rediscover previous information.

With the rapid evolution in technology and rich information collections, users demand more robust search applications to help with solving exploratory problem tasks. Further, in this chapter, the second feature presented in White's and Roth's list of search application features, (”offer facets and metadata-based result filtering”), and the faceted search will be described in more detail, as this thesis focuses on the use of facet categories to filter on a set of results.

2.3 Faceted classification

Before being able to search effectively in a collection of information, the information needs to be organized in a structured and meaningful manner. One method, as mentioned, is faceted classification. A short definition is that a facet, also called dimension or feature type, is an orthogonal set of categories or a category hierarchy relevant to the collection, that is used as a way to classify information (Tunkelang, 2009). To give some examples in the context of bibliographic library collections, some meaningful facets can be ”Author”, ”Form of work” and ”Language”. Each facet, in turn, contains a set of values or labels, also called categories, that will be used as the preferred terminology throughout this report, like ”Novel”, ”Short story” and ”TV adaptation” in the facet for ”Form of work”.

2.4 Faceted navigation

When a set of faceted hierarchies are defined and categories assigned to items in a collection, the resulting interface is known as faceted navigation. Navigation in the context of facets includes the act of viewing or navigating through an information collection with specific attributes or characteristics as a result of the categorization of different facets. Creating queries in an information collection that a user has limited knowledge about, can be challenging. The option to navigate can, in this case, help users get an overview of the information collection available and promote new ideas from the exposure of the information content.

The objective of faceted interfaces is to allow users with flexible navigation, provide previews, organize results in a meaningful way, and support both the expansion and refinement of the search (White and Roth, 2009). Much of the success of faceted navigation derives from the early work of query preview, a method for supporting dynamic querying where a visual display of the available results for the current query are displayed to the user in real time while the user is forming a search query (Plaisant et al., 1999). Faceted navigation has adapted to this key feature of giving a preview of the total available results for each category. This is usually accomplished by presenting a numeric value wrapped in parentheses on the right side of each category label to specify the total results matching each category. Letting the user know the result size of category collections, can support user’s decision making of the next steps, and avoid empty result set as this technique of faceted navigation only lets the user choose within categories with results available.

Faced navigation systems as discussed helps end users in getting an overview of the information collection available, provides a fast exploration of relevant information to user needs and avoids the zero-hit paradigm. However, developing faceted navigation systems comes with a cost. Much effort is needed in designing and deciding which facets are the most meaningful for a collection, how should the facets be displayed to the user without overwhelming or cause confusion to the user and in addition, both existing and future contents must have metadata applied consistently for each facet. Furthermore, the faceted navigation adds interaction cost by introducing the users with multiple options to comprehend and manipulate.

2.5 Faceted search

Finally, the key topic for this master thesis is presented in this section. This chapter has mentioned that traditional search systems, often only provides a simple search field for the user to enter. Further, a more powerful mechanism of faceted navigation was presented letting the user easily and rapidly traverse through a large collection of information with specified characteristics. Faceted search is, simply put, a search interface supporting both free text search and faceted navigation, **figure 2.1**. Faceted search inherits the aspect of allowing the user to explore the available collection, avoiding zero-hit and at the same time allowing the users to add additional queries when the users have a more confident idea of their information needs to retrieve a smaller collection with a higher precision.

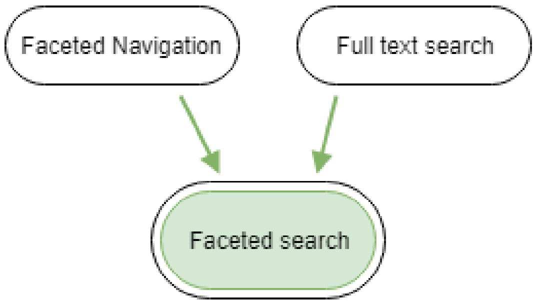


Figure 2.1: Faceted search

2.6 Search Interfaces

An extensive selection of books and research materials can be found describing the algorithms behind search engines and information retrieval systems. Only in the last decade, computer scientist and nonprofessionals have come to realization of the enormous importance of usability and UI design (Hearst, 2009). Shneiderman, an expert in the field of effective human-computer interface design, also believes that improving the search design can solve most of the challenges mentioned in the beginning of this section, for instance leading to a greater number of positive outcomes, shortening the learning times, improving user effectiveness, reducing errors, increasing the likelihood for the user to keep on using the system, and raising user satisfaction (Shneiderman et al., 1997).

The topic of UI design raises, however, a new set of challenges. The classical challenge in designing UIs is considering the diversity of end users. Users using a search application for the first time needs time to get an overview and understanding of all the possible user interactions (buttons, inputs, information, etc.), infrequent users need structure, familiar landmarks, reversibility, and safety during exploration, while experienced users want more shortcuts to speed repeated tasks.

In addition, in many cases when users are looking for information, the information is to be used in completing a more complex task. Information searching requires, therefore, a high level of concentration of the users and any distractions or complexity in the UI should be avoided. On the other hand, search applications have developed more complex functionality to exploit the rich and complex information. Consequently, UI designers face the challenges in creating simple and intuitive UI, that at the same time does not compromise the full potential of the implemented functionality.

All search interfaces can be improved to have decent or good UIs. However, inconsistency across different search interfaces can cause slower user performance, uncertainty, mistaken assumptions, and in worst case failures to find relevant documents. With the growing topic of UI design, Shneiderman has offered design principles for building effective search interfaces:

- **Strive for consistency**: using the same terminology, instructions, layout, color, and fonts across the search interface increases the user performance and satisfaction.
- **Provide shortcuts skilled users**: users who know exactly what to look for should be able to directly retrieve back the result with few clicks.
- **Offer informative feedback**: the search application should inform the user how to search for information, what happens after the search and why the system act accordingly, to help users better understand its function and to learn to use the search application as efficiently as possible.
- **Design for closure**: allowing users to make multiple choices can be disorienting. The system should, therefore, let the user know clearly when they have reached an end of the searching process.
- **Offer simple error handling**: error messages should be specific, constructive, and no more technical than necessary.
- **Support user control**: most users prefer a sense of control, and systems should let users initiate action, monitor progress of long searches, to always let the user feel in control.
- **Reduce short-term memory load**: during users search process, the users' goal may change. Instead of letting the users themselves keep track of every step of the search process, the system should provide mechanisms to reduce the short-term memory load to help users focus more on the task at hand. Examples are offering histories to permit easy reversal of action and presenting all vital information on the same page, preventing the need for sliders.

Chapter 3

Implementation and design

This chapter aims to present the implemented system with a description of the system's functionality, the technologies used and design choices made for this project. Section 3.1 starts with an introduction to the data structure in BIBSURF's information collection. Section 3.2 and 3.3 explains the functional requirements for BIBSURF and presents in more detail, the communication flow from the perspective of an end user in a use case diagram and the flow from the perspective of the system in sequence diagrams. Section 3.4 describes the implemented system and the technologies used. Finally, the filtering algorithm along with the design are described in section 3.5.

3.1 Data model

This section aims to explain the FRBR's Group 1 entities: item, manifestation, expression, and work where the BIBSURF's information collection is formed after. **Figure 3.1** shows a diagram of the relationship between the different entities, where the image is taken from the FRBR's final rapport (IFLA, 1998). The diagram indicates that a work *is realized through* one or more (expressed with a double arrow in diagram) than one expression and in turn, an expression *is the realization* of one and only one (single arrow) work. An expression may be embodied in one or more than one manifestation and so on.

The FRBR considers a **manifestation** as the physical form that bears the same intellectual or artistic content and physical features of a given work. Whereas an **item** is a single exemplar of a manifestation. **Figure 3.2** shows how the textual information in a manifestation is displayed in the implemented system.

An **expression** is the intellectual or artistic realization of a work in the form of alphanumeric, musical, or choreographic notation, sound, image, object, movement, etc., or any combination of forms. **Figure 3.3** shows the display of an expression listing the manifestations that embody the expression in the single tab named *Editions*.

Lastly, a **work** is the distinct intellectual or artistic creation. **Figure 3.4** shows the display of a work where each expression that is a realization of the work, is displayed in its own tab. **Figure 3.5** displays the work as a tree structure.

To put everything in perspective with an example, consider Shakespeare's Romeo and Juliet. The example is a representation of a *work*. A publication of the work in the form of a novel is the *expression*, and the different editions of the novel are the *manifestations*. One exemplar of an actual physical book is the *item*.

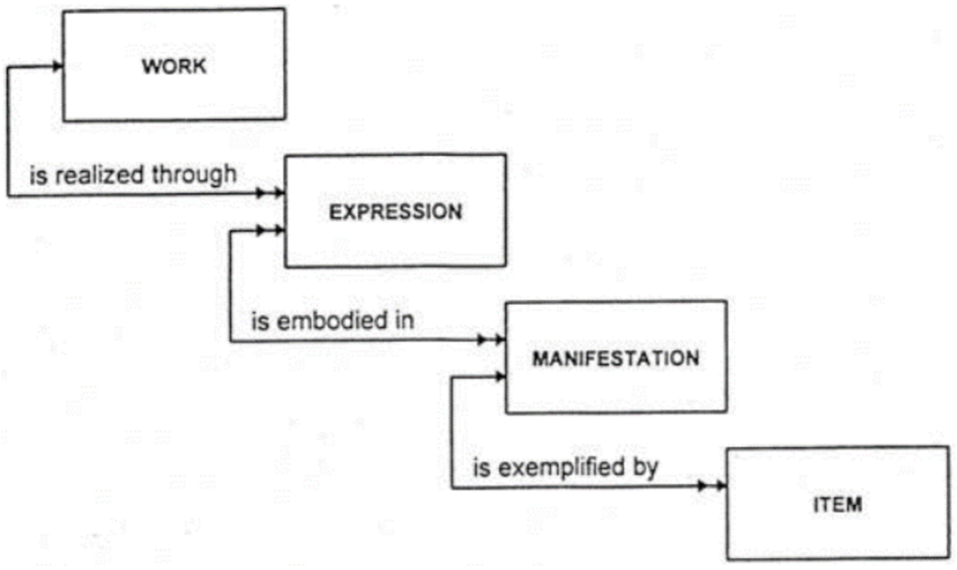
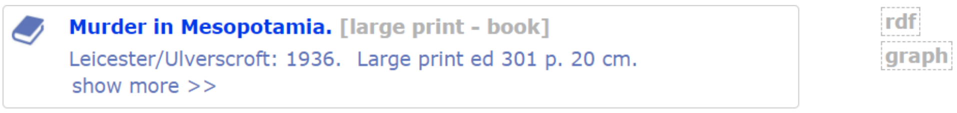


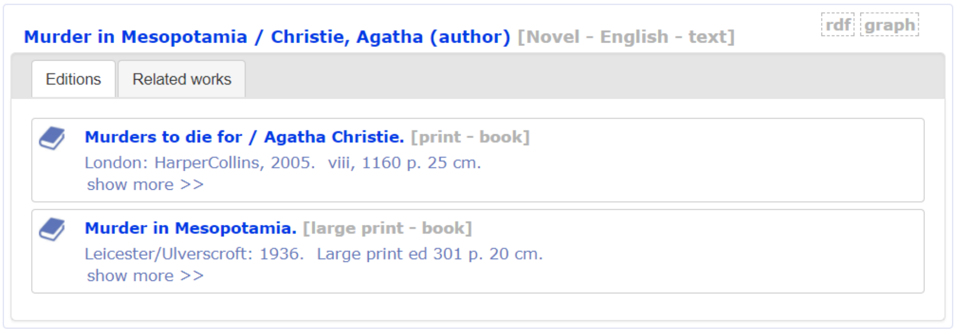
Figure 3.1: FRBR: Group 1 Entities and Primary Relationships



Murder in Mesopotamia. [large print - book]
Leicester/Ulverscroft: 1936. Large print ed 301 p. 20 cm.
show more >>

rdf
graph

Figure 3.2: Display: Manifestation


Murder in Mesopotamia / Christie, Agatha (author) [Novel - English - text]

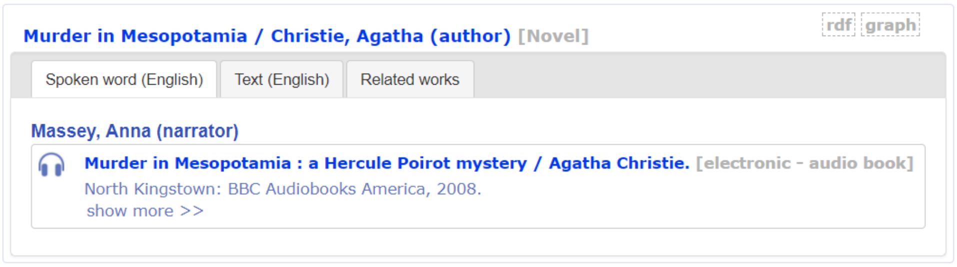
rdf graph

Editions | Related works

Murders to die for / Agatha Christie. [print - book]
London: HarperCollins, 2005. viii, 1160 p. 25 cm.
show more >>

Murder in Mesopotamia. [large print - book]
Leicester/Ulverscroft: 1936. Large print ed 301 p. 20 cm.
show more >>

Figure 3.3: Display: Expression


Murder in Mesopotamia / Christie, Agatha (author) [Novel]

rdf graph

Spoken word (English) | Text (English) | Related works

Massey, Anna (narrator)

Murder in Mesopotamia : a Hercule Poirot mystery / Agatha Christie. [electronic - audio book]
North Kingstown: BBC Audiobooks America, 2008.
show more >>

Figure 3.4: Display: Work

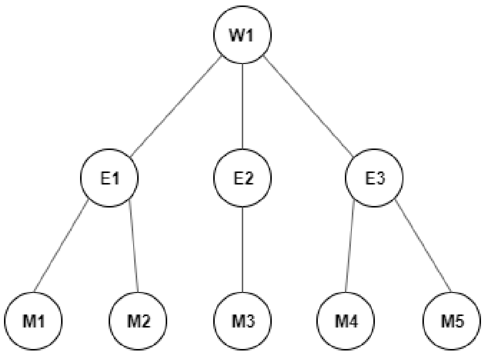


Figure 3.5: FRBR graph of a work

3.2 Functional requirements

Table 3.1 and **3.2**, specifies the functional requirements for BIBSURF:

Id	Description
FR1	The search field lets the user enter one or more terms or phrases to do a new search on the system when ”Submit” button or enter key is clicked and the categories already selected from a previews search gets reset.
FR1.1	A ”breadcrumb” button is created representing the query for the search.
FR1.2	Facets with categories are updated to let the user refine the result set with a set of categories.
FR1.3	The categories in a facet is ordered in the by highest total results and alphabetically by name if two or more categories have the same total available results.
FR11	Only the first two facets are open.
FR1.4	Maximum 25 results units are retrieved back to the user.
FR1.5	The first tab of the two first results units are open for expression or work display.
FR1.6	A ”Next” button is created at the bottom of the results to get the next 25 results if the search has more result units.
FR2	User can choose matching criteria for *match*, *display*, *ranking* and *collection* by clicking on the options from the dropdown boxes, and the search gets reset.
FR3	When a user clicks on a ”breadcrumb”, the query is removed from the search along with the ”breadcrumb” and the search gets reset.
FR4	User can view the RDF data in XML format using the RDF link, and view the graphical visualization of the RDF data using the graph link for each result unit.
FR5	The expression view includes an ”Edition” tab to view the list of manifestations of the expression.
FR6	The work view includes multiple tabs of expressions of the work, grouped by type and language.
FR7	Expressions or works includes an additional ”Related works” tabs if the result unit includes related works.
FR8	The user can view a list of contents when ”show more” button is clicked in manifestations.

Table 3.1: FRs: Search for results

Id	**Description**
FR9	User can view or hide the categories in a facet by clicking on the ”+”/”-” button.
FR10	A facet shows a maximum of 6 categories and user can view and hide the rest of the categories by clicking on the ”more”/”less” button.
FR11	User can select a category to refine the search result by clicking on the checkbox and a new search is performed.
FR12	User can remove all selected categories by clicking on the ”Clear all filters” button placed at the top of the facets.
FR13	Search performed with AND logic retrieves results matching all the selected categories (conjunction/ ”and” between all the categories, e.g. category1 *and* category2 *and* category3).
FR14	OR logic retrieves results matching at least one of the selected categories (disjunction/ ”or” between all the categories, e.g. category1 *or* category2 *or* category3).
FR15	ANDOR logic retrieves results matching at least one of the selected categories in each of the facets with categories selected (”or” between categories in the same facet and ”and” between facets, e.g. category1 *and* (category2 *or* category3).
FR15.1	Only one facet where the first category is selected by the user, let the user select multiple categories.
FR15.2	When only one category is selected and the user selects the second category from a different facet, then the facet where the second category got selected, lets the user select multiple categories.
FR16	The expression and work display have the option to view the ”sub-tree” of the results and only the parts of the expression or work that matches the selected categories are shown.
FR17	The first number after each category shows the total hit of the category for a search.
FR18	When categories are selected, each category may include a second number with a ”+” symbol and parentheses around used in ”or” and ”andor” option to show the total result available but not shown in the result set.
FR19	For ”andor” option when categories are selected in multiple facets, categories shows the first number and only a ”+” symbol in parentheses without any number.
FR20	For each category, the category text is set to a maximum of 26 letters to fit into a single line.

Table 3.2: FRs: Refine search result by facet categories

3.3 Component tree

Figure 3.6 presents all the hierarchy structure of the components or JavaScript classes created for this project.

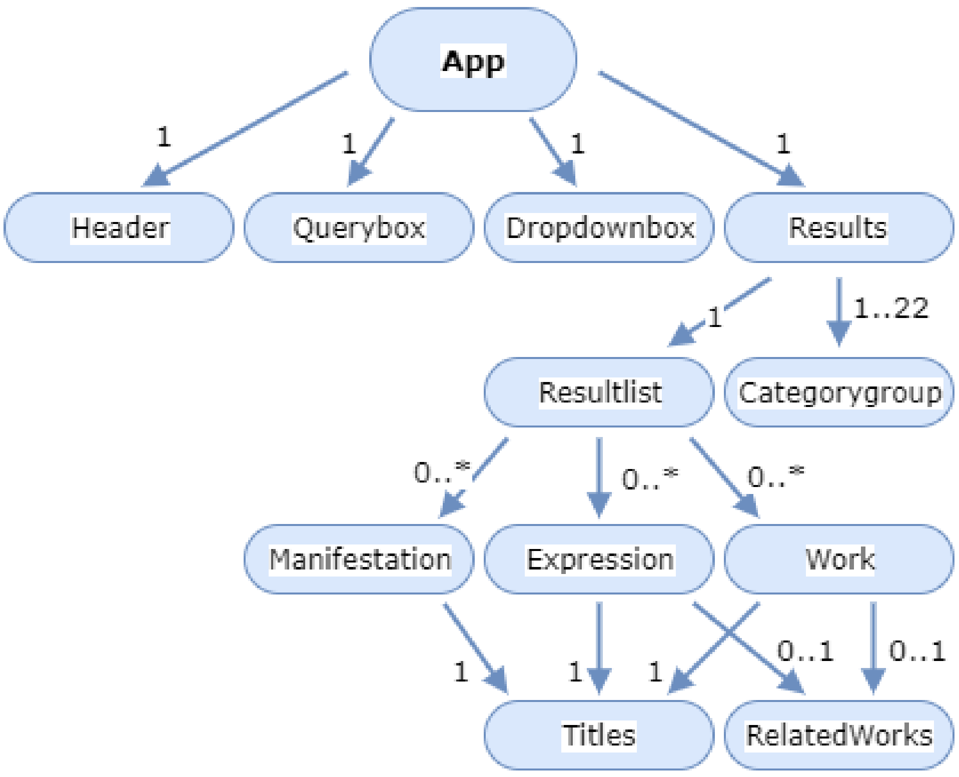


Figure 3.6: Component tree

- **App**: the App is the container component fetching all the data from the server, manipulates and stores the data in states and receives query input and category checkbox, dropdown and button changes and handles them accordingly.
- **Header**: displays the title of the application.
- **Querybox**: receives user input and sends it back to the App component to perform a search and in addition displays the query input as breadcrumbs.
- **Dropdownbox**: the App creates four Dropdownbox components to display all the options for a match, ranking, display and collection as the query parameters to send back the App component.
- **Results**: passes the data from the App component to its child components Categorygroup components and a Resultlist component, receives events from its child

component and send it back to the parent App component to handle, displays the category crumbs and in addition displays the view of the form box for choosing the filtering option on the search result.

- **Categorygroup**: or ”facet”, displays all the available categories in the respective facet and sends events to its parent if user check or unchecks the checkboxes of the categories.
- **Resultlist**: displays the results with the Manifestation, Expression, or Work component(s) according to the display type.
- **Manifestation**: displays textual information for a manifestation and includes an expandable list of contents.
- **Expression**: displays two tabs, with Manifestation component(s) in the first tab and a RelatedWorks component in the second tab if the expression has any related works associated with it.
- **Work**: displays information in multiple tabs where each expression with a unique type and language property are grouped in one tab displaying an Expression component and an additional RelatedWorks component as a tab if the work has any related works.
- **RelatedWorks**: displays a list of related works.
- **Title**: the Manifestation, Expression, and Work components display a Title component as a part of the heading displaying the names of persons and their respective titles or roles (e.g. director, actor, author, etc.).

3.4 UML

Unified Modeling Language (UML) is a standard way to visualize the design of a system with different abstractions. A visualization can be used to bring out the bigger picture by hiding or masking details or to focus more on different aspects of the system.

3.4.1 Use Case

The use case diagram in **figure 3.7** shows how the end user can interact with the system. From the home page, the user is presented with a single search field and the only interaction with the system is to add a query to perform a search. After a search, the user is presented with a list of the results. From this point on the user can choose to remove the existing search query, change the search criteria, including the change of display, match type, ranking, collection, filter option and filter with sub-tree, and lastly, adding categories to the search to refine the result set.

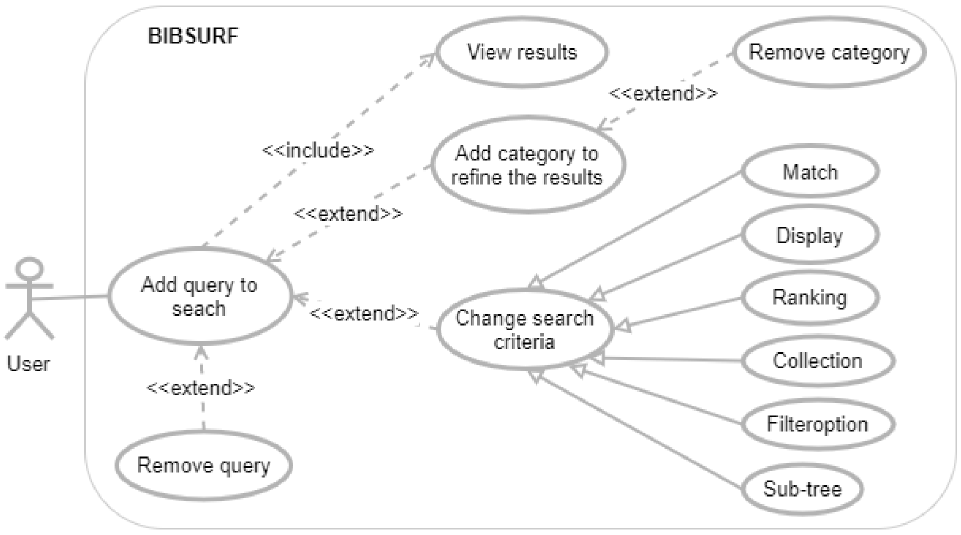


Figure 3.7: Use Case Diagram: Search application BIBSURF

3.4.2 Sequence diagram

The sequence diagrams show the interaction flow on the client side.

Figure 3.8 shows the sequence diagram when a user enters a query. When the user enters the query, the child component sends the query up to the parent component App. The App component handles the query and calls a get request to the server. When the response is returned back to the App component, it then sets the returned results as the new state by calling setState(). After the state is updated, the new state is passed down to the child component Results and further sends the results to its child component Resultlist which triggers a render to update the view. Now the user can view the results of the search.

The sequence diagram in **figure 3.9** follows the same flow where the child component reads the user interaction, sends data to the root component App to handle, the App component calls a request, updates the state when the response is returned back and sends the new states to the child component to update the view.

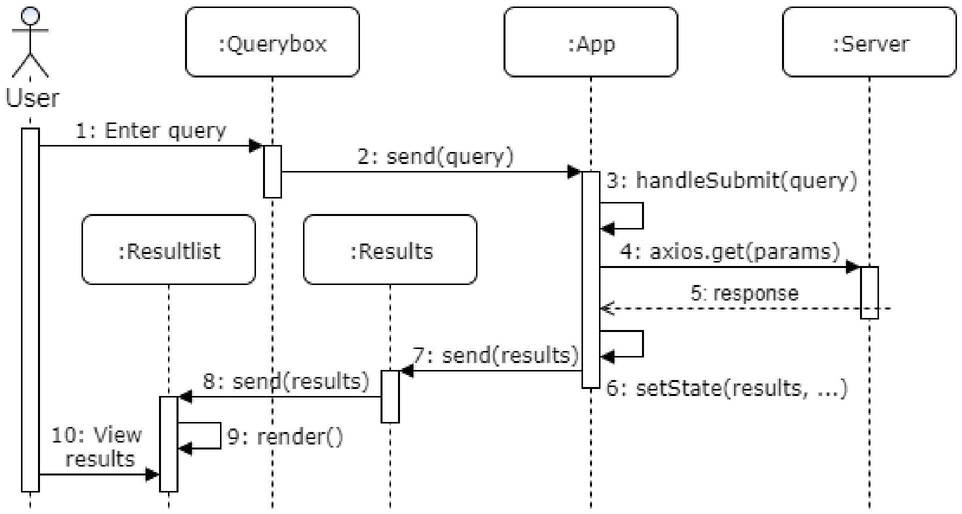


Figure 3.8: Sequence Diagram: Enter a search query

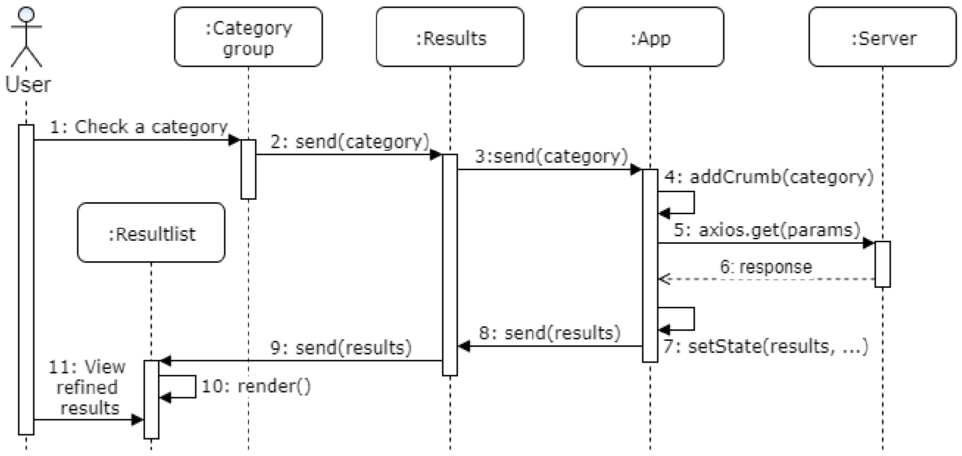


Figure 3.9: Sequence Diagram: Choose a category to refine the results

3.5 Implementation

In order to keep track of when and what changes have been made and allowing for the easy undoing of changes, the open source version control system, Git[1], was used. A repository was made and uploaded to Github[2], a web-based hosting service. The source code can be found using this link: `https://github.com/trondaal/bibsurf`[3].

Figure 3.10 shows an abstract system architecture of the BIBSURF application, using eXist, an open source native XML database, at the server side, React js with its local state management system at the client side and axios to fetch data from the server to the client side. All development was developed using the text editor Atom[4].

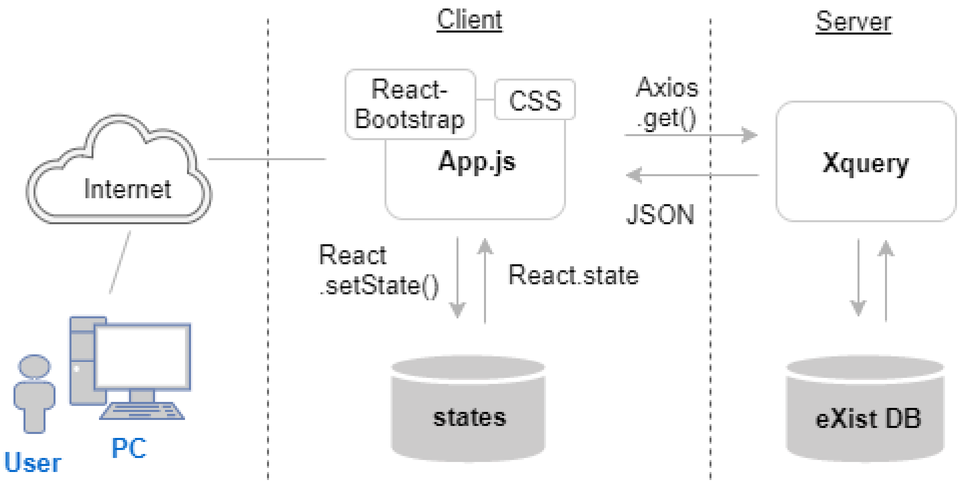


Figure 3.10: Abstract system architecture

3.5.1 React

React js[5] is a JavaScript (JS) library for building interactive user interfaces for web, mobile and desktop platforms. React has in recent years grown significantly in popularity and are used by several big companies[6] like Facebook[7], Instagram[8] and Airbnb[9]. React only deals with the view layer and additional libraries needs to be used to handle things such as data flow, routing, authentication and etc.

[1]`https://git-scm.com/`
[2]`https://github.com/`
[3]The source code was published here to be available after the student has graduated
[4]`https://atom.io/`
[5]`https://reactjs.org/`
[6]`https://github.com/facebook/react/wiki/Sites-Using-React`
[7]`https://www.facebook.com/`
[8]`https://www.instagram.com/`
[9]`https://www.airbnb.com/`

From the specialization project (Fung, 2017), a comparison of the framework Angular[10] and the libraries jQuery[11] and React, was made to propose a suitable structure for building the client side of the search application, BIBSURF. To summarize the choice for using React, was based on the lightweight library that supported building a UI that could achieved the functional requirements for this system, at the same time allowing for scalability for future development. React has additionally a well-written documentation and a growing community for collaboration and support. **Table 3.3** shows a comparison between the framework and libraries:

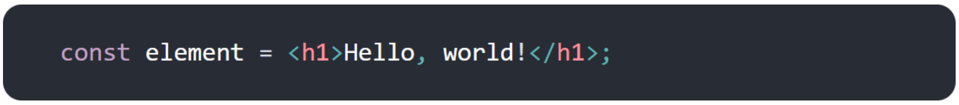
	React	**JQuery**	**Angular**
License	BSD	MIT	MIT
Language	JSX	JavaScript	JavaScript
Data binding	One-way	One-way	Two-way
DOM	Virtual	Regular	Regular
Component based	✓	x	✓
Design for testability	x	x	✓
Templating directives	x	x	✓
Form validation	x	x	✓
Routing	x	x	✓
Learning curve	Low	Low	High

Table 3.3: JavaScript frameworks and libraries for building UIs

The fundamental concepts of React includes building components, use of states, props and PropTypes and the use of a virtual Document Object Model (DOM). Components are basically a JS class with the *render* method to create the view for the user by outputting HTML-content, which is implemented using JavaScript XML (JSX) or JS code. JSX is simply a JS extension incorporating HTML-like syntax elements. **Figure 3.11**[12] taken from reactjs.org shows an example of a variable declaration with a JSX element. States in React's local storage. To change states, the asynchronous method setState() is used, and a re-render is triggered. And this.state is used for accessing the data. Data passed from a parent component to a child component is passed with props. PropType in React is used for type checking, thus for each component, it can check the expected props it will receive. This can help catching bugs in the system, by requiring for specific props using .isRequired attribute. Lastly, the HTML DOM is a mechanism for presenting and interacting with the view presented to the user. Elements of HTML become nodes in the DOM, and dynamic changes on the contents of a page is done by manipulating the DOM, usually done with the use of JavaScript. The virtual DOM is a virtual representation of a DOM object, in other words, a copy of the real DOM. Virtual DOM cannot change the content on a page but is much faster to update than the regular DOM. React uses the virtual

[10] `https://angularjs.org/`
[11] `https://jquery.com/`
[12] `https://reactjs.org/docs/introducing-jsx.html`

DOM, and whenever a re-render is triggered, the virtual DOM gets updated. Now React can a do a ”diff” computation to compare the last virtual DOM with the updated one, React can then find the exact virtual DOM elements that have been changed and can update only those changes to the regular or ”real” DOM to update the content of the page. Using the virtual DOM to find out what has changed before updating the changes to the DOM, is much faster than doing DOM manipulation directly on the DOM.



```
const element = <h1>Hello, world!</h1>;
```

Figure 3.11: JSX syntax example

React components can be categorized into two categories: containers/stateful components and stateless components. Stateless components simply have one task to display data received from its parent React component. Containers, on the other hand, are the ones doing the work of fetching data from external sources, feed data it to its child components and receive input and events to initiate actions. The components implemented in this project is structured in a hierarchy with one container component, App, as the root node. **Figure 3.6** explained earlier in this chapter summarizes all the components created for this project.

3.5.2 Create-react-app

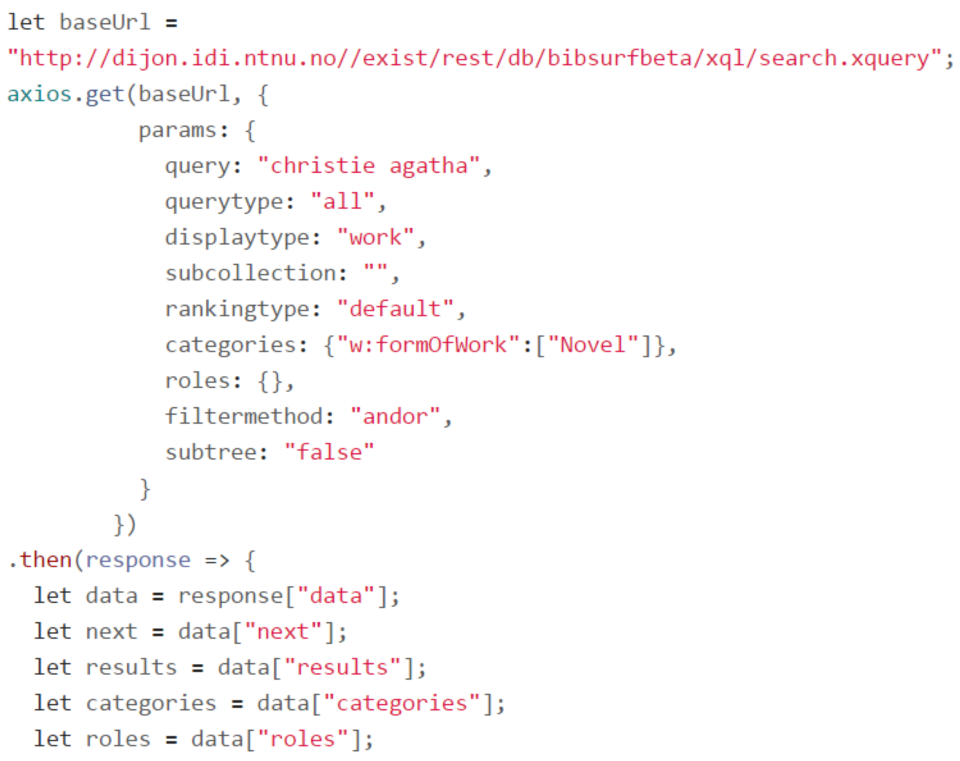
This project uses the module create-react-app[13] (CRA), a starter project developed by Facebook that lets developers start developing React applications without needing to set up configuration. The advantages of using this package are that it does not acquire any time-consuming configurations setup from the developer before creating a running React application in a couple of minutes. But in turn, it does not support flexible customizing of the configurations, in other words, it does not allow the developer to configure some additional dependencies. The main tools used in CRA are:

- **Webpack**: a bundler tool to collect all the code and dependencies in one JavaScript file.
- **Babel**: used to compile ES6, a newer JavaScript update, to work with older browsers.

[13] `https://github.com/facebook/create-react-app` Support for building single-page React applications, and offering a modern build setup with no configuration.

3.5.3 Axios

Axios'[14] GET method was used to fetch the results from the server. The method accepts a url and optionally some specified parameters. The query is constructed by the terms or phrases from user input, separated by a space. And the other parameters are from the selected options in the Dropdown components. **Figure 3.12** shows an example of a request for getting the first 25 results:



```
let baseUrl =
"http://dijon.idi.ntnu.no//exist/rest/db/bibsurfbeta/xql/search.xquery";
axios.get(baseUrl, {
          params: {
            query: "christie agatha",
            querytype: "all",
            displaytype: "work",
            subcollection: "",
            rankingtype: "default",
            categories: {"w:formOfWork":["Novel"]},
            roles: {},
            filtermethod: "andor",
            subtree: "false"
          }
        })
.then(response => {
  let data = response["data"];
  let next = data["next"];
  let results = data["results"];
  let categories = data["categories"];
  let roles = data["roles"];
```

Figure 3.12: Code snippet: an axios's get call

The response is a JSON object, mapping a value for next, results, categories and roles, and are manipulated and stored on the client side as states:

- *next*: the url to get the next 25 hits.
- *results*: an array list of the results of the first 25 hits.
- *categories*: an object including all the categories and their respective count of the total result.
- *roles*: an object including all the roles and their respective count of the total result.

[14] `https://www.npmjs.com/package/axios`, A promise based HTTP client for the browser and node.js

3.5.4 React-bootstrap

The BIBSURF UI, originally used jQuery UI[15] elements to build parts of the UI. However, the UI showed some incompatibility when the UI was re-written in React-components. Therefore the React-Bootstrap[16] framework was used to build some of the application's UI elements as the researcher had worked with the Bootstrap[17] library before. The project uses the following React-bootstrap components: Button, FormControl, Well, Panel, Collapse and Col. The FormControl is used to create the input field for query search and the dropdown boxes for the user to select the query options. React-Boostrap has a Tab component available, however, it was not possible to hide the tab contents so one tab needs to be open initially. For that reason, it was decided to create a custom tab feature for this project since the React-bootstraps' Tab component did not meet the needs of this project. To emulate the tabbed feature, the Button component was used to represent a tab, the Collapse component was used to view the tab information and lastly, the Panel was used to put the all the content with basic border and padding. In addition, a new state value *activeTab* was added, storing the currently active tab to get the correct information to show in the Collapse component, and a second state *open* to decide when to open and close the Collapse component. When the user clicks on the tab which is currently the active tab, the information in the Collapse component will be hidden (state *open* set to false), or if the user clicked on a tab that is not active the information will be changed accordingly (state *open* set to true if it was false). Lastly, the Col component is React-bootstrap's grid system for placement of responsive elements and this project uses Col to create a neat placement of the search input field and submit button on the opening home page.

3.5.5 Uuid

Uuid[18] generates cryptographically strong and random id's. Uuids were applied as the key property of child components. Keys help React identify which elements that have been changed when updating the virtual DOM. Each item in the BIBSURF collection includes a unique id, and this id could in practice have been used as the key. However, each item may be listed multiple times in the application, thus having the same key for multiple elements. It was therefore decided to create new unique id keys with the uuid library to assign each child component.

[15] `https://jqueryui.com/` jQuery UI is a set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library.

[16] `https://react-bootstrap.github.io/` The most popular front-end framework, rebuilt for React.

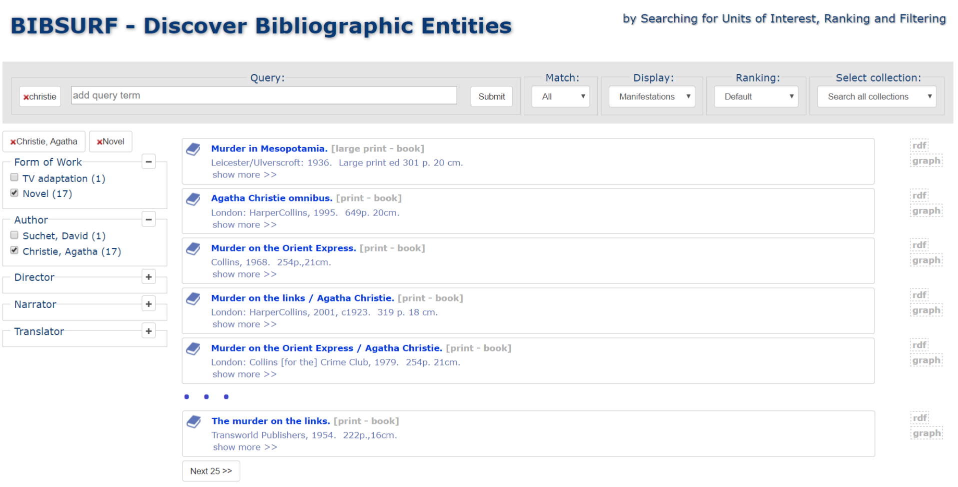
[17] `https://getbootstrap.com/` Bootstrap is an open source toolkit for developing with HTML, CSS, and JS

[18] `https://www.npmjs.com/package/uuid` Simple, fast generation of RFC4122 UUIDs.

3.6 User Interface

The final implemented UI and an explanation of the design choices will be presented in this section. **Figure 3.13** shows the implemented UI from the author's specialization project (Fung, 2017) and **figure 3.14** shows the final UI implemented. The changes made in the UI design includes the following:

- an adjustment in the design for ”and” logic,
- design for ”or” and ”andor” logic,
- a form box for selecting the filter option,
- removal of the breadcrumbs for the selected categories,
- a button for removing all selected categories,
- a button for retrieving the next 25 search results, and
- a home page.



BIBSURF - Discover Bibliographic Entities

by Searching for Units of Interest, Ranking and Filtering

Query: xchristie | add query term | Submit
Match: All
Display: Manifestations
Ranking: Default
Select collection: Search all collections

xChristie, Agatha xNovel

Form of Work
- TV adaptation (1)
- [✓ Novel (17)]

Author
- Suchet, David (1)
- [✓ Christie, Agatha (17)]

Director

Narrator

Translator

Murder in Mesopotamia. [large print - book]
Leicester/Ulverscroft: 1936. Large print ed 301 p. 20 cm.
show more >>

Agatha Christie omnibus. [print - book]
London: HarperCollins, 1995. 649p. 20cm.
show more >>

Murder on the Orient Express. [print - book]
Collins, 1968. 254p.,21cm.
show more >>

Murder on the links / Agatha Christie. [print - book]
London: HarperCollins, 2001, c1923. 319 p. 18 cm.
show more >>

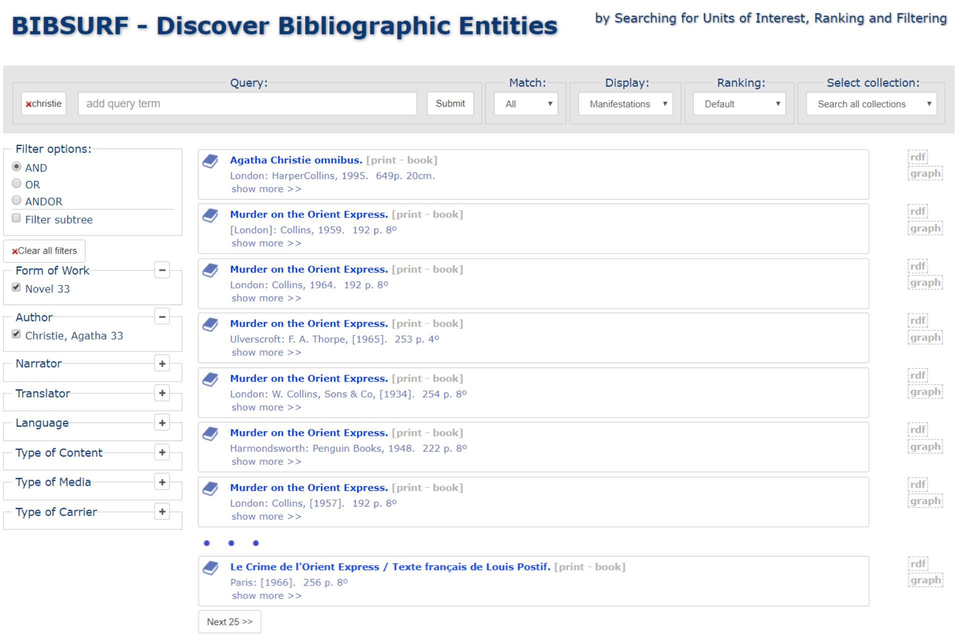
Murder on the Orient Express / Agatha Christie. [print - book]
London: Collins [for the] Crime Club, 1979. 254p. 21cm.
show more >>

• • •

The murder on the links. [print - book]
Transworld Publishers, 1954. 222p.,16cm.
show more >>

Next 25 >>

Figure 3.13: UI Specialization project



BIBSURF - Discover Bibliographic Entities

by Searching for Units of Interest, Ranking and Filtering

Query: xchristie | add query term | Submit
Match: All
Display: Manifestations
Ranking: Default
Select collection: Search all collections

Filter options:
- [✓ AND]
- OR
- ANDOR
- Filter subtree

xClear all filters

Form of Work
- [✓ Novel 33]

Author
- [✓ Christie, Agatha 33]

Narrator

Translator

Language

Type of Content

Type of Media

Type of Carrier

Agatha Christie omnibus. [print - book]
London: HarperCollins, 1995. 649p. 20cm.
show more >>

Murder on the Orient Express. [print - book]
[London]: Collins, 1959. 192 p. 8º
show more >>

Murder on the Orient Express. [print - book]
London: Collins, 1964. 192 p. 8º
show more >>

Murder on the Orient Express. [print - book]
Ulverscroft: F. A. Thorpe, [1965]. 253 p. 4º
show more >>

Murder on the Orient Express. [print - book]
London: W. Collins, Sons & Co, [1934]. 254 p. 8º
show more >>

Murder on the Orient Express. [print - book]
Harmondsworth: Penguin Books, 1948. 222 p. 8º
show more >>

Murder on the Orient Express. [print - book]
London: Collins, [1957]. 192 p. 8º
show more >>

• • •

Le Crime de l'Orient Express / Texte français de Louis Postif. [print - book]
Paris: [1966]. 256 p. 8º
show more >>

Next 25 >>

Figure 3.14: UI Final

3.6.1 Form box

Figure 3.15a) shows the form box created to let the user choose a filter option to refine the search result. The form box includes a radio button group for choosing one of the filter option for ”and”, ”or”, and ”andor”, and a checkbox at the bottom to choose if the result should show the complete result items or only the parts satisfying the selected categories, for the expression and work display. The form box was created following the same design as the form boxes for each facet as shown in **figure 3.15b)** as a comparison.

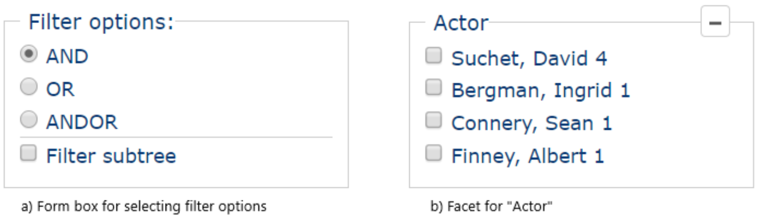


Figure 3.15: UI Form box for selecting filter options and categories in a facet

3.6.2 Breadcrumbs

BIBSURF uses breadcrumbs to represent all the search queries performed by the user. When submit is entered for a search query, a button with the search query as the label is created at the left side of the search field as shown in **figure 3.16**. Each breadcrumb has a red X associated to indicate to the user that the breadcrumb will be removed when the breadcrumb is clicked on.

From the original BIBSURF UI, breadcrumbs, placed on top of the facets, were also used to represent or remove the categories selected to refine the search results as shown in **figure 3.13** or in appendix B. The breadcrumbs were only allocated a small section of space in UI surface. As users chose several categories to filter the search results, the list of breadcrumbs grew accordingly and pushed the placements of facets further to the bottom of the page. As the breadcrumbs took up space, it was decided to remove the breadcrumbs. In the new implementation, users need to click on the checkboxes for each category to add or remove the filter to the search result. In addition, a button was added above the facets to provide the functionality to remove all the selected categories for the current search.

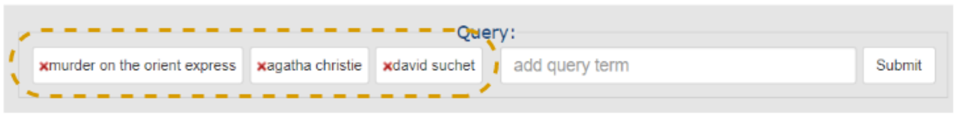


Figure 3.16: UI Breadcrumbs

3.6.3 Home page

Test users, from the user study described in the next chapter, felt that the initial starting page was missing a home page. For this reason, a simple home page was created during the work of this thesis, as shown in **figure 3.17**. The home page consists of a search field to enter a search query, a button to submit the query, the title for the search application and a background. The simplicity of the design was inspired by the home page of the online university library at NTNU[19]. The image used as the background for the home page was taken from a photo competition at NTNU for international students (spring 2015), that were open to all international students that studied at NTNU in the autumn semester 2014 and/or spring semester 2015. The image can be found using this link: `https://www.facebook.com/NtnuInternational/photos/a.586435994832890.1073741848.167232473419913/588911511252005/?type=3&theater`.

> ”Petals of Dandelion, this photo was captured in spring 2013 in Flakk near Trondheim during my stay at NTNTU.”
>
> — Hammad Majeed, master’s degree student from Pakistan.

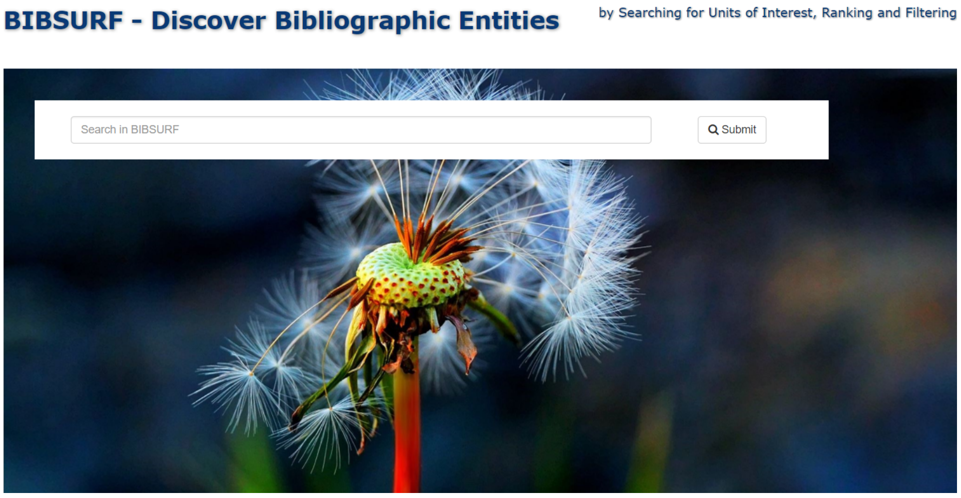

BIBSURF - Discover Bibliographic Entities
by Searching for Units of Interest, Ranking and Filtering
Search in BIBSURF
Submit

Figure 3.17: UI Homepage

[19] `https://www.ntnu.no/ub`

3.7 Faceted filtering

This section aims to describe the filtering method and visualization used when users select categories to refine the search result.

3.7.1 And sub-tree selection algorithm

This chapter aims to present the idea of the filtering algorithms used to filter the information collection, by explaining the algorithm for the "and" selection with sub-tree filtering of works. **Figure 3.19** visualize the decomposition of a FRBR graph to a set of branches.

Figure 3.18 describes the procedural pseudo-algorithm to the query. The algorithm simple checks for each entity level (work, expression, manifestation) and returns back all the branches where no categories or all the selected categories for the current entity type matches with the entity for all entity levels.

```
foreach work in resultset {
  if (no work-categories are selected)
  OR (all selected work-categories matches work) {
    foreach expression in work.getExpressions()
      if (no expression-categories are selected)
      OR (all selected expression-categories matches expression)
        foreach manifestation in expression.getManifestations()
          if (no manifestation-categories are selected)
          OR (all selected manifestation-categories matches manifestation)
            add work, expression, manifestation to set of selected entities
  }
}
```

Figure 3.18: Procedural pseudo-algorithm

- *resultset*: search result (graphs with work as node)
- *categories*: set of selected criteria each as a tuple of (entitytype, category, value) where entitytype is the type of entity that the selected criteria applies to (e.g. *author* is a work-level category, language is an expression-level category as shown in **table 3.4**)

The algorithm for "or" selection with sub-tree filtering of works, works similar to the algorithm for the "and" selected described above. The "or" selection uses "at least one" instead of "all" in the "and" selection in the matching criteria. The next sections describe the implemented visualizations of the faceted filtering.

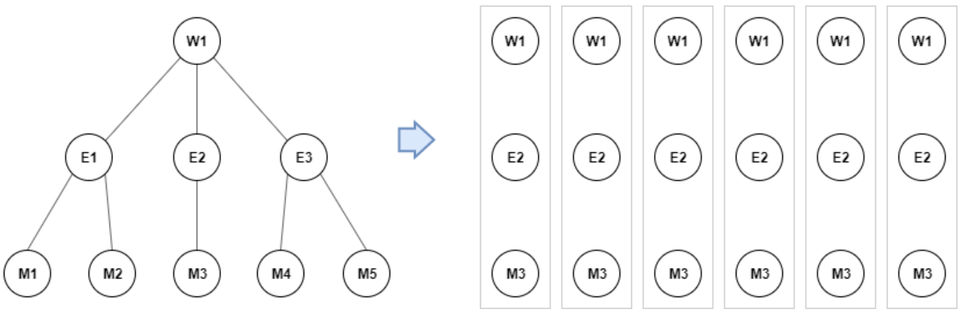


Figure 3.19: Decomposing a FRBR graph to a set of branches

Facets	**Entity type**
Form of Work, Artist, Author, Compiler, Composer, Contributor, Director, Interviewee, Interviewer, Lyricist, Screenwriter	Work
Abridger, Adaptor, Conductor, Narrator, Performer, Translator, Language, Content Type	Expression
Type of Media, Type of Carrier	Manifestation

Table 3.4: Entity type for each facet

3.7.2 Conjunction and

From the server, the response of the query request returns a JSON object with the available categories, an example shown below:

```
{
    "categories" : {
        "m:carrierType" : {
            "book" : 18,
            "audio␣book" : 3,
            "video" : 5
        },
        "w:formOfWork" : {
            "Literary␣critisism" : 2,
            "Autobiography" : 2,
            "TV␣adaptation" : 5,
            "Biography" : 2,
            "Novel" : 12
        }
    }
}
```

The label of each category is used as the key in the JSON mapping the value of the total hit of results. All the available categories with the total number of hits, retrieved from the server, are displayed the same way in all the filter options for ”and”, ”or” and ”andor” as shown in **figure 3.21a**).

The initial UI from the specialization project was only using ”and” logic as the only filtering option to refine a search. Each category is associated with a number in parentheses representing the total hit for the search query. For each selected category the total number of hit for each category gets updated and only non-empty categories are displayed to the user.

In addition to the numbers associated with the categories, categories with filter option for ”or” and ”andor” may have an additional number associated to each category, as will be discussed in the next sections. After experimenting with different placements of parentheses, it was evaluated that the most suitable design was to surround the second number in parentheses in cases where categories included two numbers for the filtering option for ”or” and ”andor”. It was therefore decided to remove the parentheses around the first number with a total hit, to keep a consistent design across the different filtering options. Accordingly, the numbers for each category in ”and” option was without parentheses as shown in **figure 3.21b)**.

3.7.3 Disjunction or

The ”or” option show all the results matching any of the selected categories. **Figure 3.21c)** shows the display of the categories for ”or” option. The first number represents the total number of hit, displayed and retrieved the same way as the ”and” option. When categories are chosen, the remaining unchecked categories may include a second number showing the total number of available hit that is not shown in the result set. From the initial search, all of the available hit on a search is presented. To get the second number, the App component stores the value from the initial search and when new categories are added to the search, the second number can be calculated by subtracting the initial values with the new values (3.1).

$$x = v_0 - v_\mathrm{N} \tag{3.1}$$

- x = the second number showing the total number of hit that is not shown in the result set.
- $\mathrm{v_0}$ = the value of total hit in the initial search.
- $\mathrm{v_N}$ = the value of total hit when categories are applied to the search result.

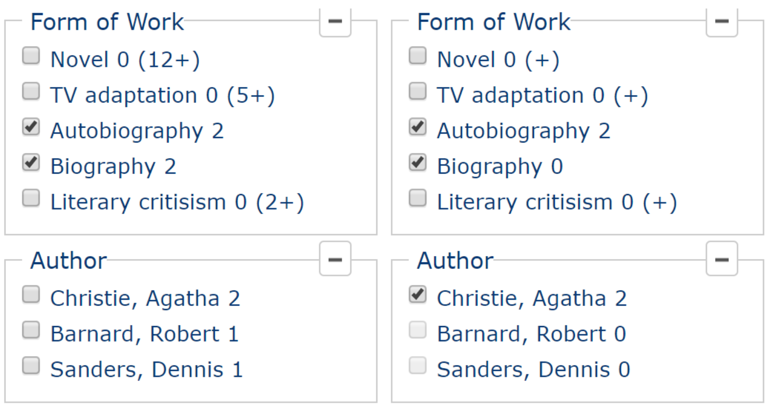
When categories with two numbers are checked, the second number would be calculated to be zero. Instead of just showing a zero, the second number gets removed if the result is zero, and the category shows the first number a with total number of hit. From the perspective of the user, when the category is checked, the second number gets merged into the first number. It was therefore decided that it was appropriate to add a "+" symbol to indicate that new results are added to the result display.

3.7.4 Andor

The "andor" option combines the "and" and "or" option and a visualization of the display is shown in **figure 3.21d)**. It allows "and" between the facets and "or" between the categories in each facet. Because of this flexibility, letting the user combine categories across multiple dimensions at the same time using a more strict selection algorithm, the probability of zero-hit is increased. To address the challenge of zero-hit, a set of rules was designed:

- Reduce the flexibility by allowing only one facet with the ability for choosing multiple categories, where the chosen facet will be referred to as the "or-box" in the context of "andor" option throughout this thesis since the categories within this facet or "box" is selected with an "or" option. The "or-box" can be chosen in two different ways:
  - The facet where the first category is clicked on is chosen to be the "or-box".
  - Experimenting with different interaction patterns, the researcher considered the scenario when a user first selects one category from a facet and then wants to select multiple categories from a second facet, e.g. the user searches for works, and selects first the category "Novel" as the "Form of Work" and further selects multiple categories in the facet of "Author". This was done by checking if only one category is selected in the first facet and when the user chooses the second category within a different facet than the first category. Then the category in the second facet is chosen to be the "or-box".
- If the user removes a category within the "or-box" while categories are selected within other facets, then the client side of the system handles this by removing all categories that are not present in the "or-box". Consider the scenario where the user chooses the categories "Novel" and "Short story", and the facet of "Form of work" is chosen to be the "or-box". The user then selects one author in the "Author" that only has a short story. Now the user removes the category for "Short story". This will lead to a zero-hit since the already chosen author only has a short story and not a novel. The current state of the system does not support validating if removing categories leads to zero-hit. It was therefore decided to handle this by unchecking categories outside the "or-box" when the user chooses to remove a category in the "or-box".

The visualization of the "or-box" follows the design of the categories in "or" option. However, the system does not support for calculating the second number for the scenario where the user has selected categories across multiple facets. So instead of showing the total number of available hit, only the "+" surrounded with parentheses is shown to indicate that the category may have more available results, **figure 3.20b)**. The idea of the "andor" option is to offer a more exploratory search process compared to the stricter option of "and". Hence, when categories are selected, the user still sees the categories without any hit to maintain the overview of the categories where the user may decide to relax its selected categories in order to explore the different categories available.



a)

Form of Work

- Novel 0 (12+)
- TV adaptation 0 (5+)
- [✓ Autobiography 2]
- [✓ Biography 2]
- Literary critisism 0 (2+)

Author

- Christie, Agatha 2
- Barnard, Robert 1
- Sanders, Dennis 1

b)

Form of Work

- Novel 0 (+)
- TV adaptation 0 (+)
- [✓ Autobiography 2]
- [✓ Biography 0]
- Literary critisism 0 (+)

Author

- [✓ Christie, Agatha 2]
- Barnard, Robert 0
- Sanders, Dennis 0

Figure 3.20: UI Andor display

3.7.5 Sub-tree

The original UI did not include the option to view the sub-tree of works or expressions. **Figure 3.19** presented with a view of the tree structure of a work. In the original UI when at least one branch satisfied with the selected categories, the ”whole” tree of the work was displayed. The option of sub-tree was developed in this project using the algorithms as presented earlier, offering the user to view only the sub-tree of works or expressions. **Figure 3.22** is an example of a work displaying the whole tree where the categories ”spoken word” and ”English” were selected. From the figure, all other expressions of the work in different languages (German, Croatian, Italian, etc.) are also displayed. **Figure 3.23** shows the same example of the work with the same categories applied, however with the option to view the sub-tree instead. In this figure, only the expression for spoken word in the language English is displayed. Categories view zero-hit are disabled as can be seen in **figure 3.20b**).

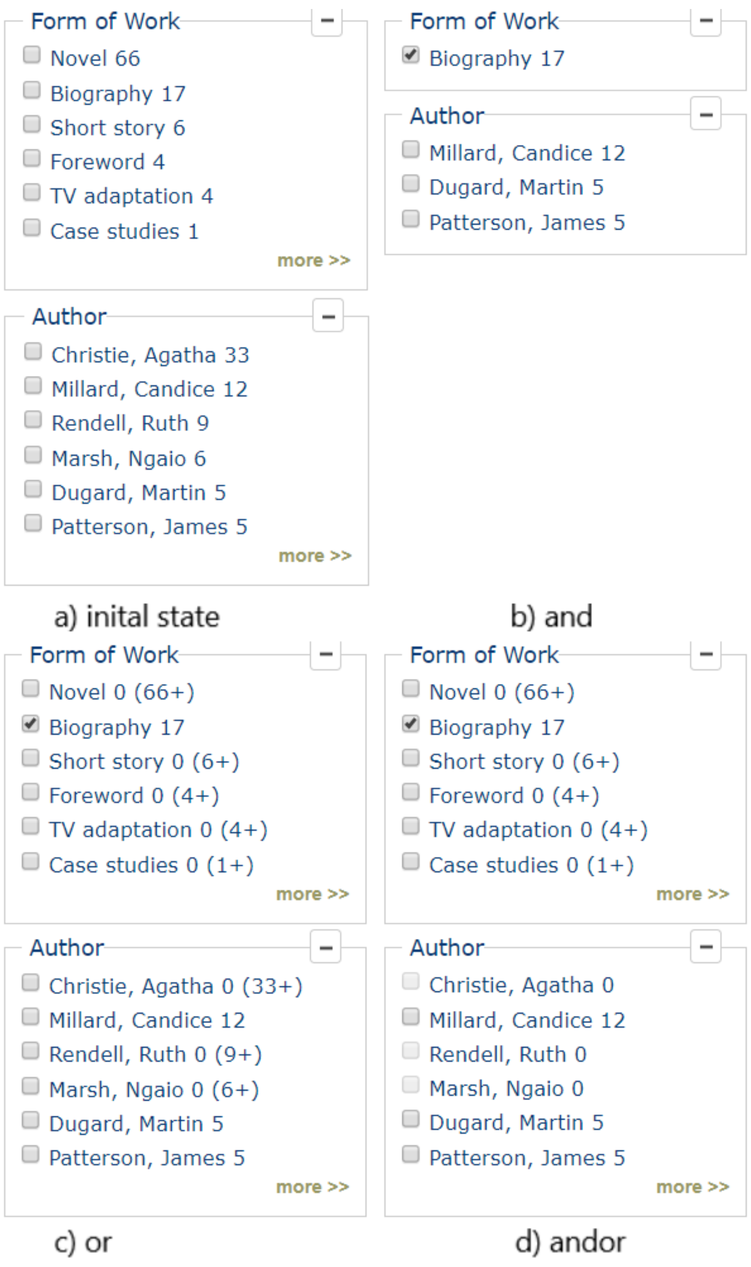
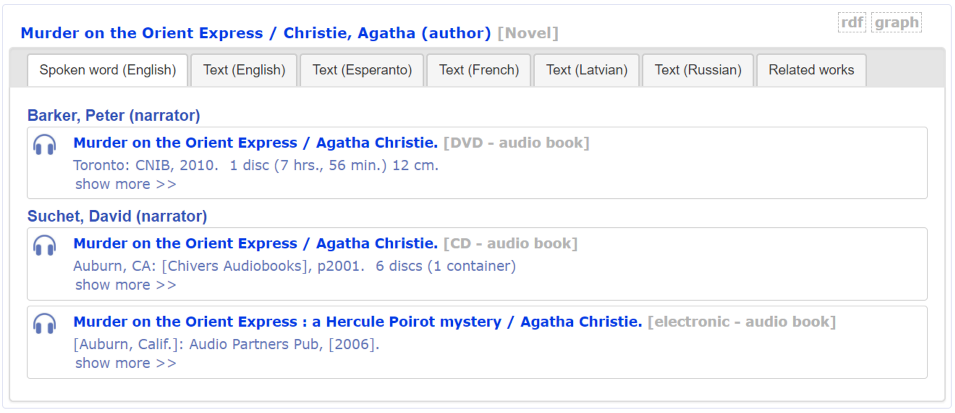


Figure 3.21: UI Categories in facets


Murder on the Orient Express / Christie, Agatha (author) [Novel]

rdf graph

Spoken word (English) | Text (English) | Text (Esperanto) | Text (French) | Text (Latvian) | Text (Russian) | Related works

Barker, Peter (narrator)

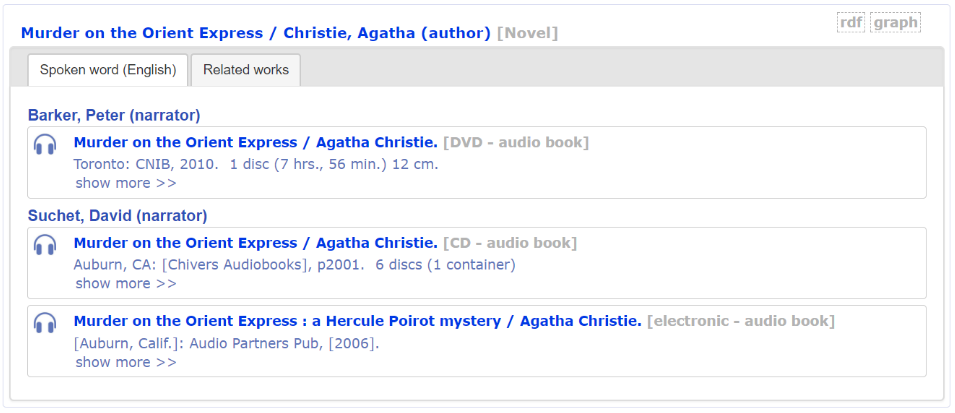
Murder on the Orient Express / Agatha Christie. [DVD - audio book]
Toronto: CNIB, 2010. 1 disc (7 hrs., 56 min.) 12 cm.
show more >>

Suchet, David (narrator)

Murder on the Orient Express / Agatha Christie. [CD - audio book]
Auburn, CA: [Chivers Audiobooks], p2001. 6 discs (1 container)
show more >>

Murder on the Orient Express : a Hercule Poirot mystery / Agatha Christie. [electronic - audio book]
[Auburn, Calif.]: Audio Partners Pub, [2006].
show more >>

Figure 3.22: Work with the whole tree displayed


Murder on the Orient Express / Christie, Agatha (author) [Novel]

rdf graph

Spoken word (English) | Related works

Barker, Peter (narrator)

Murder on the Orient Express / Agatha Christie. [DVD - audio book]
Toronto: CNIB, 2010. 1 disc (7 hrs., 56 min.) 12 cm.
show more >>

Suchet, David (narrator)

Murder on the Orient Express / Agatha Christie. [CD - audio book]
Auburn, CA: [Chivers Audiobooks], p2001. 6 discs (1 container)
show more >>

Murder on the Orient Express : a Hercule Poirot mystery / Agatha Christie. [electronic - audio book]
[Auburn, Calif.]: Audio Partners Pub, [2006].
show more >>

Figure 3.23: Work with the sub-tree displayed

Chapter 4

Experiment

This chapter describes how the user study was conducted and explores the users' understanding of the implemented filtering mechanism as described in the research questions RQ-2.1 and RQ-3 from **table 1.1** in chapter 2: "Does the user understand the design associated with the categories?" and "Does the user understand the refined result when categories are applied to the search?". This user study lets users test the implemented design, explained earlier in this report, and is divided into two different iterations. In the first test, the researcher discovered: situations where the test approach caused confusion to the user, and limitations in exploring the research questions. Hence, a second test where explored with adjustments from the feedback in the first test.

All participants participated in this user study are students at NTNU. The participants were asked to sign a form allowing the researcher to record the computer screen with audio during the test and allowing the results to be used in this report. The contract form can be found in the appendix A. The test executions use researchers' laptop and the computer screen was recorded using the Screen Recording functionality in Office's PowerPoint, an accessible and easy to use tool for the researcher. During this test, the users are presented with the term "filter" to what this project has been referring to as a category, as the users are more familiar with the term filter.

During testing, the user will be asked to think aloud and the researcher will not assist the user in solving the tasks, to avoid influencing the users' decisions or understanding of the application. Further with the screen recordings including audio, the observations can be analyzed more precisely afterward, including overlooked details in the initial observations. Additionally, the completion time of each task can be noted more precisely and eliminate any inconsistency in registering the completion time. After the test, the user is asked to answer a short survey followed by an informal interview to collect additional information.

Each filter option has different challenges in how to best present the categories in a way that it easy for the user to understand the application's functionality used in different contexts. In other words, the goal of this experiment is to see if the user is able to understand the application with the given design of the categories. **Table 4.1** below describes the challenges in creating a user-friendly design when the different filter options are applied in this experiment.

And	When categories are applied with ”and” as filter option, the user has to toggle between multiple categories to view all the right results. For example, when a user wants to get results from multiple categories in a specific facet.
Or	The filter option ”or”, increases the result set as more categories are applied. Consequently, this filter option may lose its value if the user retrieves the original result set after a lot of categories are applied.
Andor	The design chosen to display the categories for ”andor” are not conventional compared to other search applications with filters. It is therefore interesting to explore if the user observes the difference in behaviour, depending on which categories are applied first and second.
Sub-tree	When the sub-tree filtering is disabled, the users have to navigate through the tabs of different languages and content types to find the right results even though the categories for language or content type are already applied. This experiment wants to observe if the user finds this inconvenient, preferable or do not take notice of this at all.

Table 4.1: Design challenges for each filter option

4.1 Preliminary test

The first test approach, uses a similar methodology as a user study conducted earlier in Spring 2017, to be able to compare and support this research. The user study from Spring 2017 was conducted by my supervisor and his colleagues (Aalberg et al., 2017), exploring the different displays of the results and compared their ability to support different user tasks using BIBSURF. To summarize, the results indicated that the work display worked well in exploring and learning about repertoires, but made it difficult to identify specific manifestations or expressions. The expression display, in turn, worked well when looking for publications in specific languages. While the manifestation display remained consistent for the different user tasks.

In the preliminary test, a total of 6 users participated covering the different filter options: two users testing each filter option, ”and”, ”or” and ”andor”, where one user uses filtering with the option of retrieving the sub-tree of the results and the other user uses filtering retrieving the whole tree. Each user will be given the same 4 tasks to solve, described in **table 4.2** below, including the scenarios, predefined queries and the objective for each task.

Scenario 1	You have recently seen the movie ”Murder on the Orient Express”, based upon a novel by Agatha Christie, and would like to read the novel.
Query	”**Murder on the Orient Express**”
Task	Find one book of the novel available in English.
Objective	The user have to identify one edition of the novel. Using the ”and” or ”andor” option with the sub-tree filtering, the exact results is given back to the user when the right combination of categories are applied. Categories appropriate for this task are: Novel (Form of Work), book (Type of carrier), Christie Agatha (Author), English (Language).
Scenario 2	You got an assignment from school to write an essay around the life of Agatha Christie.
Query	”**Agatha Christie**”
Task	Find the available autobiographies and biographies of Agatha Christie.
Objective	This task takes advantage of the ”or” or ”andor” option to retrieve multiple results from a specific facet, in this case ”Form of Work”. The goal is to get result of both autobiographies and biographies. Categories: Autobiographies (Form of Work) and Biographies (Form of Work).
Scenario 3	You have read one version of the novel ”Don Quixote” in English and are interested in reading the novel in other languages as a way to practise your language skills.
Query	”**Don Quixote**”
Task	Find one book of the novel available in Spanish and one in Italian.
Objective	In this case the user wants books in different languages. Hence, this task exploits the ”andor” option with the sub-tree filtering option, to filter on novels at the same time allowing the user to choose multiple languages. Categories: Novel (Form of Work), book (Type of carrier), Spanish (Language), Italian (Language).
Scenario 4	You like listening to audio books and have recently liked the audio books narrated (”forteller”) by David Suchet.
Query	”**David Suchet**”
Task	Find one electronic audio book available, that are narrated by David Suchet.
Objective	David Suchet appears as author, actor and narrator. Using the ”and” or ”andor” option with the sub-tree filtering easily retrieves the exact results back to the user. Categories: Novels (Form of Work), David Suchet (Narrator), audio book (Type of Carrier), electronic (Type of Media).

Table 4.2: Scenarios in preliminary test

As the objective is to explore the intuitiveness of the presented categories, the filter option will be hidden to the user during testing, to explore the design quality of the categories. Further, to focus on the users' interaction with the categories to refine a search, the user is given predefined display, query, and tasks. Lastly, only the work display will be used for all test scenarios, to be able to explore the filtering with sub-tree and to keep this research as consistently as possible as this research does not further explore the different display options.

The results use the following measurements to analyze the users' understanding of the filtering mechanism:

- **Description score**: users understanding of the filtering option after all tests are finished, where 5 = complete description, 4 = one element of description missing, 3 = two elements missing, 2 = three elements missing and 1 = no relevant description.
- **Success score**: successful completion of each task where 5 = complete success, 3 = partial success and 1 = no success.
- **Time**: the time needed to complete each task.

The execution of each user test followed the following steps:

1. First, the user is asked to fill in a form to let the researcher record the computer screen and audio during testing.
2. The user is then asked to pick a number between 1-6 to choose a filtering option, where each filter option is assigned a random number as shown in **table 4.3**. Hence, the user will not know which filtering option is chosen.
3. Further, the user is introduced to this research as followed: "This research explores the use of filters in search applications. Together with my supervisor, we have created a search application, called BIBSURF, where you can search in a small collection of novels. I would like you to use this system to solve 4 different tasks with different scenarios. For each task, type in the given query and choose your filters to minimize the search result. During the test I want you to think aloud and try as best as you can to

    - express your first impressions,
    - explain what you want to do and
    - what you expect the application to do.

    As my research is focusing on the use of filters, I want you to pay attention to the numbers and the plus symbol behind each filter and explain to me what the numbers and plus indicate and what the filters do to the results after you have finished all the tests. For each task, stop whenever you feel you have found your answer. The results retrieved from this application is structured in a hierarchy."
4. After the introduction, the user is given the tasks to look at and start whenever the user is ready.

5. When the user finishes with all the tasks, the user is given a short survey form to answer described in **table 4.4** with both multiple choice questions and open-ended questions to respond with their own answers.
6. Finally, there will be an informal interview between the researcher and the user to collect additional information.

	And	Or	Andor
Full tree	5	2	3
Sub-tree	4	6	1

Table 4.3: Random numbers for deciding the filter method

Question	Type	Description
Q1	Multiple choice	I use filters to refine a search on web sites that provides search with filters. (eg. Zalando, eBay, Oria)
Q2		How familiar are you with search in bibliographic catalogs? (eg. Oria)
Q3		How easy was the application to use?
Q4		Would you like to use this system again?
Q5		How confident did you feel using the application?
Q6	Open-ended	Describe how the filtering mechanism works.
Q7		How did you interpret the numbers behind the filters?
Q8		If you could change one thing about the design of the filters, what would it be and why?
Q9		What did you like the most abut the design of the filters and why?
Q10		Other comments about the search application or the test in general?

Table 4.4: Survey questions

The objective of the survey questions is to collect additional data and get an awareness about the users and users impressions and understanding of the implemented application. Questions Q1-Q2 asks the users about their previous experiences with using filters in search applications and questions Q3-Q5 asks about users impression of the implemented application. Further Q6-Q7 asks the user to describe the implemented filtering mechanism and finally, questions Q8-Q10 asks for feedback on the design and other comments about the user study. The survey were given to the users using SurveyMonkey[1], an online survey software, to collect the data.

[1]`https://no.surveymonkey.com/`

4.2 Observations

The overall impression of the test process from the participants in the preliminary test were good and from the results, the tasks were successfully completed without any significant difficulties. However, the tasks used in the test plan limited the exploration of users' impression and understanding of the filtering mechanism.

Most of the time, the user were focused on completing the given task, to find the different manifestations. Additional, the work display where received as overwhelming in the beginning, with a lot of elements displayed at the same time. For this reason, users were focusing on the results and used most of the time to be accustomed to the display of the results and paying no or little attention to the categories.

During testing, the search toolbar served as a distraction: Users tried to type in their own query or extend the predefined queries when they were allowed to type in the queries in the search field. Further, when users were exploring the application, in the beginning, users tried to change the ”Match” criteria in the toolbar.

When a new search query is applied, only the first two category groups of filters and the first two works in the results are open while the rest are closed by default. The users participated did not notice this feature in the very beginning during the first three tasks. Consequently, in the first two tasks, some users only explored the available results that were initially open. Users did not notice the possible to open other facets either, since the first two facets were initially open. Hence, only the categories in the first two facets were used in the beginning.

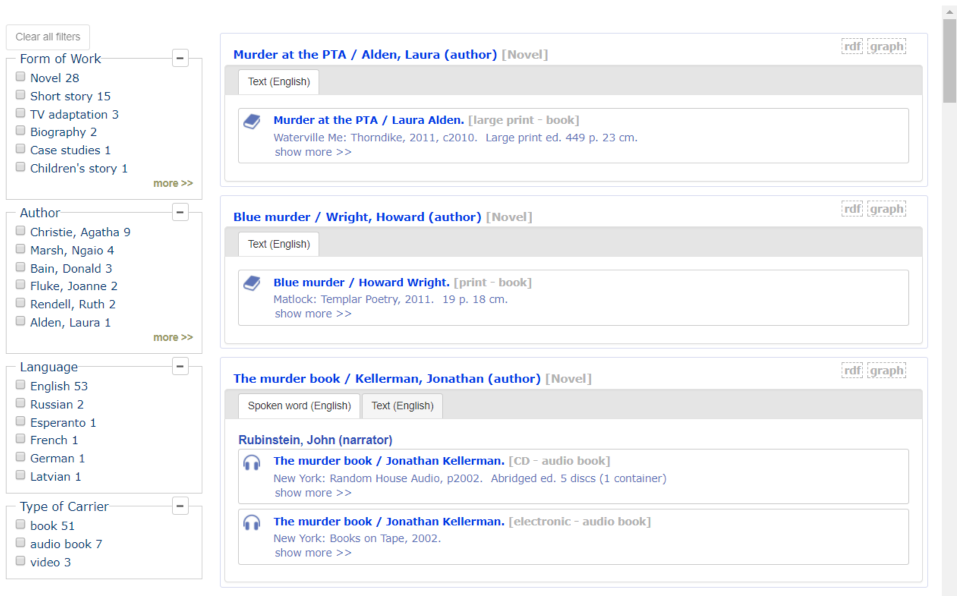
From the informal interviews, users thought that the first two facets occupies a lot of space and found it inconvenient to scroll down the list of facets to find the categories the user were looking for. One example, where the facet of ”Language” was placed almost at the bottom of the list. Additionally, users do not understand which categories are included in the facets of ”Type of Content”, ”Type of Media”, or ”Type of Carrier” and force the user to open each facet to find out its content.

The queries used in the preliminary test were also too specific, returning only a small set of results. Users were, therefore, able to identify the right manifestations nearly immediately after a new query search. Further, when the users got more familiar with the tabs on each work, the tabs were used to look for works in specific languages instead of using the categories to filter for language. In the last task users only uses one or two categories to find the right manifestations.

4.3 Final test

The observations from the preliminary test revealed for potential improvements, thus a new test plan was designed for a final test. The changes in the test plan for the final test includes:

- further customization of the tasks for this experiment described in **table 4.5** including:
  - letting the user use all three filter options with only one scenario,
  - use the query ”murder” to get a long list of results and,
  - suggest filters / categories for the users to use,
- and a customized UI for the users during testing as shown in **figure 4.1** including:
  - hiding the toolbar and unused facets and
  - having all the facets and tabs in the results to be initially open.



Clear all filters

Form of Work
- Novel 28
- Short story 15
- TV adaptation 3
- Biography 2
- Case studies 1
- Children's story 1

more >>

Author
- Christie, Agatha 9
- Marsh, Ngaio 4
- Bain, Donald 3
- Fluke, Joanne 2
- Rendell, Ruth 2
- Alden, Laura 1

more >>

Language
- English 53
- Russian 2
- Esperanto 1
- French 1
- German 1
- Latvian 1

Type of Carrier
- book 51
- audio book 7
- video 3

Murder at the PTA / Alden, Laura (author) [Novel] rdf graph

Text (English)

Murder at the PTA / Laura Alden. [large print - book]
Waterville Me: Thorndike, 2011, c2010. Large print ed. 449 p. 23 cm.
show more >>

Blue murder / Wright, Howard (author) [Novel] rdf graph

Text (English)

Blue murder / Howard Wright. [print - book]
Matlock: Templar Poetry, 2011. 19 p. 18 cm.
show more >>

The murder book / Kellerman, Jonathan (author) [Novel] rdf graph

Spoken word (English) Text (English)

Rubinstein, John (narrator)

The murder book / Jonathan Kellerman. [CD - audio book]
New York: Random House Audio, p2002. Abridged ed. 5 discs (1 container)
show more >>

The murder book / Jonathan Kellerman. [electronic - audio book]
New York: Books on Tape, 2002.
show more >>

Figure 4.1: UI in final test

The scenarios created for the user study earlier in Spring 2017 worked well when the objective was to explore the display of the result set. From the observations, the new scenarios created for this experiment worked poorly in achieving the objective of this experiment. Since the scenarios created for this experiment followed the same structure as the scenarios earlier in Spring 2017.

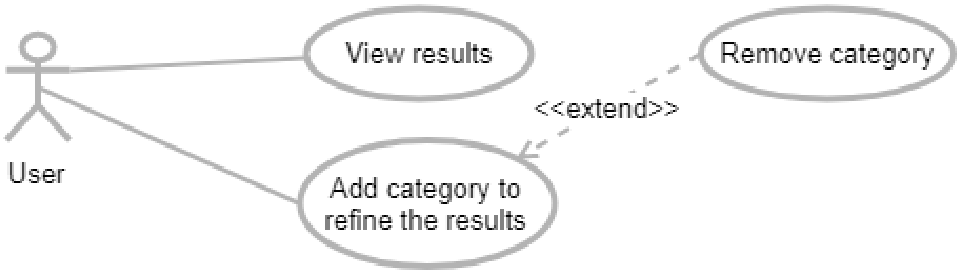


Figure 4.2: UML: Use case in final test

From the experiences from the preliminary test, the users only focused on one assignment at a time. Hence, the focus was towards identifying the manifestations asked in the tasks. Furthermore, users used most of the time to understand the display of the results during the test instead of analyzing the behaviour of the application when categories were applied. Finally, when users were asked to describe the filtering mechanism after the test, the users could only give a brief description. Therefore, in the final test, it was decided to change to directly ask the users to describe the functionality when categories are applied as the actual task of the test instead of asking the user to identify specific manifestations. Now, the users will have to pay attention to the behaviour of the application when categories are applied to each filter option.

When users got familiar with the categories and the display of the results, the remaining tasks felt repetitive without gaining a new understanding of the filtering mechanism. Users were therefore in the final test only given one scenario all the three different filter options. Additionally, each user will use the filter options in different order. To be able to observe if users will be influenced by the previous filter option when working with the next filter option. The order in which filter option the user is to be presented with was selected randomly. The users were asked to choose a number representing the order in which filter option to be used, **Table 4.6** shows the different orderings. The option to show the sub-tree will be alternated for each user, where the first user has the option of retrieving the sub-tree, the second user retrieves the whole tree, the third user with sub-tree and so on.

The queries used earlier in Spring 2017 were designed to return a small set of results and aims to let the users describe the presented display of the results. For this experiment, the main goal is to let the user explore the use of categories. As the test was created following the same flow as in the user study in Spring 2017, the queries were too specific for this experiment when users were only asked to identify specific manifestations. As a consequence, users did not apply any categories or only used a small set of categories before the users could identify the manifestations. In the final test, the query ”murder” was used to get a larger set of results. This was decided to prevent the users from spotting the manifestations that were asked for in the scenario immediately. As a consequence, applying categories to the search becomes essential to be able to find the correct information at a faster pace, and motivates the user to use the categories to refine the results. As in real search applications, usually, have a larger database retrieving a long list of results even

Scenario	You are interested in works by the author Agatha Christie and would like to discover her collection of novels. You are additionally a language enthusiast and would like to discover in which languages the novels are available in. You have made a query in the library and retrieved a list of results.
Filters	Suggested filters to use are **Novel** in ”Form of work”, **book** in ”Type of Carrier” and **Christie, Agatha** as the ”Author”. Additionally, use **English, French** in ”Language”.
Task	For each filter method: Explain what each filter you apply does to the result set and what the numbers behind the filters represent.
Objective	As the query ”Murder” returns a large set of results, the use of categories becomes essential. Using ”and” as the filter option returns all the results matching all the categories applied. The ”or” option, however, expects the user to carefully choose the categories since adding more categories increases the retrieved results. In this case, when the category ”Christie Agatha” in ”Author” is chosen, the user can utilize the numbers behind other categories to get an overview of what is included in the results that match the selected category. Suggesting the users to choose categories for multiple languages, exploits the last filter option, ”andor”. The users can choose multiple languages with ”or” logic in the facet of ”Language”, at the same time refine the results from categories in other facets of the ”and” logic.

Table 4.5: Scenario used in final test

A-O-AO	A-AO-O	O-A-AO	O-AO-A	AO-A-O	AO-O-A
1	6	5	2	4	3

A = And, O = OR, AO = Andor

Table 4.6: Random numbers for deciding the order of filter method to be used

with specific search queries.

From the observations in the preliminary test, users needed some time to be accustomed to the UI when users were allowed to interact with the entire application. The toolbar became a distraction and users wanted to change the ”Match” criteria or add an additional query to the search. Further, the users needed to scroll through the list of categories to find suitable categories to apply on the results. Hence, the toolbar and facets that were not used in the final test were hidden to focus on the design of the relevant categories. All the categories and the first tab on each work in the results were set to be initially open to keep the UI consistent as some users did not realize this feature in the beginning. In addition, the users were not informed of the query used, to prevent distracting the users from the actual task.

The execution of the final test follows the steps in the preliminary test. However, instead of asking the users to describe the answers of the tasks orally, the users are asked to write down the answers on a piece of paper. Writing down the answers challenges the users to formulate a more specific and clear answer to the tasks.

After the test, the user is asked to answer a short survey, which includes a subset of the survey given in the preliminary test. In the final test, a customized UI was used, therefore questions Q3-Q5, questions about the impression of the application, where excluded in the survey for the final test. Further, questions Q6-Q7 were excluded as they were used as the tasks during testing. The survey given after the final test includes the questions Q1-Q2 and Q8-Q10.

Chapter 5

Discussion

A summary of the end results and conclusions are presented in this chapter.

5.1 Final observations

5.1.1 And

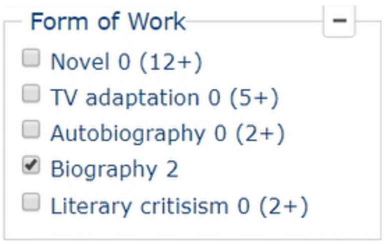
Most of the users were familiar with using filters on different applications and were used with a numeric value for each category to indicate the total available results. Hence, the users did not have any difficulties in working with this filter option. However, especially users in the preliminary test who did not test the different filtering options expressed a preference for having the first numbers in parentheses.

5.1.2 Or

The tasks both in the preliminary test and the final test were designed to utilize the category ”Novel” as it retrieved back a large set of results. This was useful when using the filter option for ”and” or ”andor” where the user could narrow down the results by applying additional categories. In ”or” filtering, when users choose the ”Novel” category first, then almost the same results as the initial search are displayed. Most users had difficulty in noticing any changes when more categories were applied and tried to look for an update button. However, the users who did notice the behaviour thought the design was intuitive and interesting as most of the users had not seen similar approaches. In addition, a few users expected the filtering mechanism to work as ”and”, and were surprised and confused when they did not get what they expected.

5.1.3 Andor

The filter option ”andor” is the most complex filter option implement. The user tests indicated that users struggled the most to understand the complete functionality of this filter option. The categories in some facets are disjoint, thus each category does not have any works in common. Therefore when users choose its first category within a disjoint facet, the other categories will be associated with a zero as shown in **figure 5.1**. In this case, some users needed some time to understand the purpose of the number zero displayed and further considered it pointless. Therefore, the users would rather prefer that the numbers remained the same as in the initial search. Moreover, some users realized immediately that the behaviour of the first category filters inside the first category group acted as an "or" logic. In this case, users thought that the results would only get larger when more filters are applied. Users would therefore often apply only one or two filters and removing selected filters before applying new ones. Further, when categories within multiple facets have been selected, the facet allowing the user to selected multiple categories got confusing when the second number disappeared and instead only displayed with the "+" symbol. Lastly, a user tends to choose all the categories in mind at the same time before looking at the remaining available categories. This leads to the users overlooking the changes in "andor" option when the user clicks on the second category in a different facet.


Form of Work

- ☐ Novel 0 (12+)
- ☐ TV adaptation 0 (5+)
- ☐ Autobiography 0 (2+)
- ☑ Biography 2
- ☐ Literary critisism 0 (2+)

Figure 5.1: UI disjoint categories in a facet

5.1.4 Subtree

All users retrieving the whole tree of the works, expected the tabs to disappear when they applied categories for "Language" and "Type of Content". This behaviour was expressed as strange and users suggested that the system could at least have changed the active tab to match the corresponding category selected. However, the mechanism for the tabs was received well. Most users thought that the display of the tabs in each work was displayed neatly and therefore did not mind that the tree was displayed.

5.2 Result scores

Filter option	Description score	Scenario	Success score	Time in seconds
AND	5	1	5	72
		2	5	90
		3	5	73
		4	5	63
AND w/sub-tree	1	1	5	88
		2	5	206
		3	5	82
		4	5	98
OR	5	1	5	147
		2	5	165
		3	5	128
		4	5	233
OR w/sub-tree	5	1	5	283
		2	3	42
		3	5	60
		4	5	141
ANDOR	3	1	5	69
		2	5	33
		3	5	39
		4	5	72
ANDOR w/sub-tree	3	1	5	94
		2	5	120
		3	5	61
		4	5	64

Table 5.1: Scores in preliminary test

#	Statement	Option	%
Q1	I use filters to refine a search on web sites that provides search with filters. (eg. Zalando, eBay, Oria)	Always	0.0
		Most of the time	83.3
		Almost half of the time	0.0
		Some of the time	16.7
		Never or almost never	0.0
Q2	How familiar are you with search in bibliographic catalogs? (eg. Oria)	Very familiar	0.0
		Quite familiar	33.3
		A little familiar	66.7
		Not at all familiar	0.0
Q3	How easy was the application to use?	Very easy	0.0
		Quite easy	83.3
		A little easy	16.7
		Not at all easy	0.0
Q4	Would you like to use this system again?	Very likely	16.7
		Likely	50.0
		A little likely	33.3
		Not at all likely	0.0
Q5	How confident did you feel using the application?	Very confident	16.7
		Quite confident	66.7
		A little confident	16.7
		Not at all confident	0.0

Table 5.2: Multiple choice in preliminary test

	Filter option	Description	Time in seconds
Whole tree	and	5	176
	andor	3	457
	or	5	318
Sub tree	or	5	456
	and	5	148
	andor	4	477
Whole tree	or	5	335
	andor	3	221
	and	5	379
Sub	andor	3	577
	and	5	357
	or	3	523

Table 5.3: Scores in final test

#	Statement	Option	%
Q1	I use filters to refine a search on web sites that provides search with filters. (eg. Zalando, eBay, Oria)	Always	0.0
		Most of the time	100.0
		Almost half of the time	0.0
		Some of the time	0.0
		Never or almost never	0.0
Q2	How familiar are you with search in bibliographic catalogs? (eg. Oria)	Very familiar	0.0
		Quite familiar	50.0
		A little familiar	50.0
		Not at all familiar	0.0

Table 5.4: Multiple choice in final test

5.3 Conclusions

Based on guidelines for building a React application, this project successfully developed a dynamic search application meeting the functional requirements described in chapter 3. This project contributed to the implementation of different filtering strategies, the implementation of paginating the results and storing local states for a rapid and efficient update for opening and hiding tabs in the results. The relatively small project made it manageable to use React's local state management system using this.state and this.setState() to update the view. However, in a growing application, the state management system can slowly become chaotic with local states. Redux and Mobx are two state management systems that can be considered when BIBSURF reaches the face of growth.

The user study conducted in this thesis revealed some obstacles in the UI design for providing a positive experience to the user when using the filtering mechanism created in this project. First-time users needed some time to be accustomed to the UI and thought particularly the view of the results were overwhelming in the very beginning. In many cases, the users did not notice the change in the results and got confused when the categories applied did not seem to do anything to the results. Some users thought the titles for each work displayed too much information, particularly when the title of the work is quite long combined with multiple roles associated with the work. Additionally, the titles use similar blue color as hyperlinks, and users expected these to be clickable.

Luckily, the user study also indicated that the search application BIBSURF was well received. Apart from the overwhelming view in the beginning, users still thought the presented UI of the application was structured in a neat and organized way especially when they got more familiar with the UI. Users thought the filters were easy to use, well structured and thought that the previews of the total number of hit were useful. Users reacted positively when the design was similar to other filtering systems they have used and acted with similar behaviours. Most people had to spend some time understanding the design of the categories, but after being accustomed to the design, the users found it useful when they realized how it worked providing them with new and interesting filtering options. However, much work is still needed before a design is accepted as a standard, especially design patterns that do not follow the main conventional design.

Usability testing is a time-consuming and complex study. This thesis only scratches the surface of a usability test. With the test plan provided in this study, future researchers can base their research on the work and results in this thesis.

Some future work to be considered are user studies targeting users in a specific group of demographics on a larger scale, to better spot trend. In addition, UI must be more carefully designed when attempting in designing for inconsistent views.

Bibliography

Aalberg, T., Merčun, T., Žumer, M., 2016. Bibsurf: Discover bibliographic entities by searching for units of interest, ranking and filtering. `http://doi.acm.org/10.1145/2910896.2925434` accessed: 1. february 2018. In: Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries. JCDL '16. ACM, New York, NY, USA, pp. 207–208.

Aalberg, T., Merčun, T., Žumer, M., 2017. Interactive displays for the next generation of entity-centric bibliographic models. `https://brage.bibsys.no/xmlui/bitstream/handle/11250/2468540/ICADL2017-Paper-17.pdf?sequence=2` accessed: 1. february 2018. Lecture Notes in Computer Science.

English, J., Hearst, M., Sinha, R., Medhurst, K., Yee, K.-P., 03 2002. Flexible search and navigation using faceted metadata. `http://flamenco.berkeley.edu/papers/flamenco02.pdf` accessed: 7. february 2018. Unpublished.

Ercegovac, Z., jun 2006. Multiple-version resources in digital libraries: Towards user-centered displays. Journal of the American Society for Information Science and Technology 57, 1023–1032.

Fung, M. J., 2017. Creating dynamic and interactive web applications for displaying large set of data. Unpublished.

Hearst, M., Elliott, A., English, J., Sinha, R., Swearingen, K., Yee, K.-P., Sep. 2002. Finding the flow in web site search `http://flamenco.berkeley.edu/papers/cacm02.pdf` accessed: 5. february 2018. Commun. ACM 45 (9), 42–49.

Hearst, M. A., 1999. User interfaces and visualization. In Modern Information Retrieval. Ricardo Baeza-Yates and Berthier Ribeiro-Neto. `http://people.ischool.berkeley.edu/~hearst/irbook/print/chap10.pdf` accessed: 10. February 2018. Addison-Wesley Longman Publishing Co., Inc.

Hearst, M. A., Apr. 2006. Clustering versus faceted categories for information exploration. `http://doi.acm.org/10.1145/1121949.1121983`, accessed: 1. february 2018. Commun. ACM 49 (4), 59–61.

Hearst, M. A., 2009. Search User Interfaces. http://searchuserinterfaces.com/book/ accessed: 6. February 2018, 1st Edition. Cambridge University Press, New York, NY, USA.

IFLA, 1998. Study Group on the Functional Requirements for Bibliographic Records. Functional Requirements for Bibliographic Records : final report. https://www.ifla.org/files/assets/cataloguing/frbr/frbr_2008.pdf accessed: 17. February 2018. UBCIM publications ; new series, vol. 19. Munich: K.G. Saur Verlag.

Kules, B., Shneiderman, B., Mar. 2008. Users can change their web search tactics: Design guidelines for categorized overviews. https://www.sciencedirect.com/science/article/pii/S0306457307001574?via%3Dihub accessed: 11. february 2018. Inf. Process. Manage. 44 (2), 463–484.

Music, D., 2017. Strategier og teknikker for filtrering av skeresulteter. https://brage.bibsys.no/xmlui/handle/11250/2454570 accessed: 9. February 2018. NTNU.

Nielsen, J., 2007. Breadcrumb navigation increasingly useful. http://www.useit.com/alertbox/breadcrumbs.html accessed: 13. february 2018. Nielsen Norman Group.

Plaisant, C., Shneiderman, B., Doan, K., Bruns, T., Jul. 1999. Interface and data architecture for query preview in networked information systems. http://doi.acm.org/10.1145/314516.314522 accessed: 7. february 2018. ACM Trans. Inf. Syst. 17 (3), 320–341.

Salton, G., Buckley, C., 1990. Improving retrieval performance by relevance feedback. http://www.cs.ucr.edu/~vagelis/classes/CS172/publications/jasistSalton1990.pdf accessed: 1. february 2018. Journal of the American Society for Information Science.

Shneiderman, B., Byrd, D., Croft, W. B., 1997. Clarifying Search: A User-Interface Framework for Text Searches http://www.dlib.org/dlib/january97/retrieval/01shneiderman.html accessed: 7. February 2018. Corporation for National Research Initiatives.

StatisticBrain, 2016. Youtube company statistics https://www.statisticbrain.com/youtube-statistics/ accessed: 9. february 2018. Digital Technology.

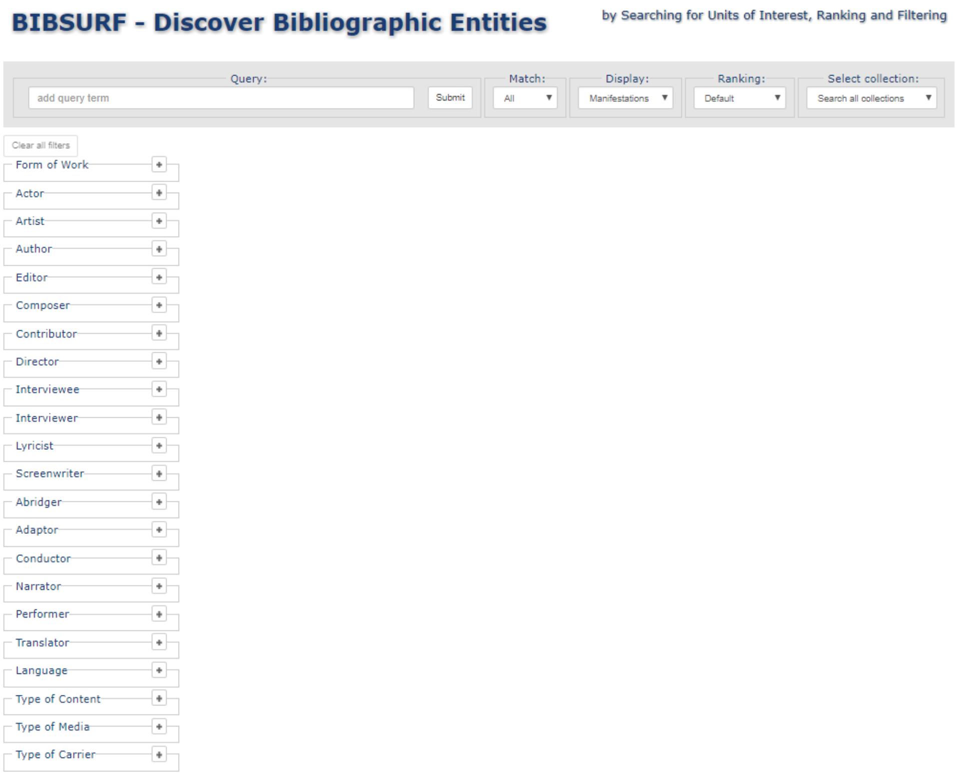
Tillett, B. B., 2003. What is frbr? a conceptual model for the bibliographic universe. https://www.loc.gov/cds/downloads/FRBR.PDF accessed: 18. february 2018. Technicalities 25 (5).

Tunkelang, D., 2009. Faceted search - synthesis lectures on information concepts, retrieval, and services. http://disi.unitn.it/~bernardi/Courses/DL/faceted_search.pdf, accessed: 23. January 2018. Morgan and Claypool Publishers.

White, R. W., Roth, R. A., 2009. Exploratory Search: Beyond the QueryResponse Paradigm. `http://www.iro.umontreal.ca/~nie/IFT6255/Books/ExploratorySearch.pdf` accessed: 30. January 2018. Morgan and Claypool Publishers.

Appendix

Appendix A
UI Elements


BIBSURF - Discover Bibliographic Entities

by Searching for Units of Interest, Ranking and Filtering

Query: add query term

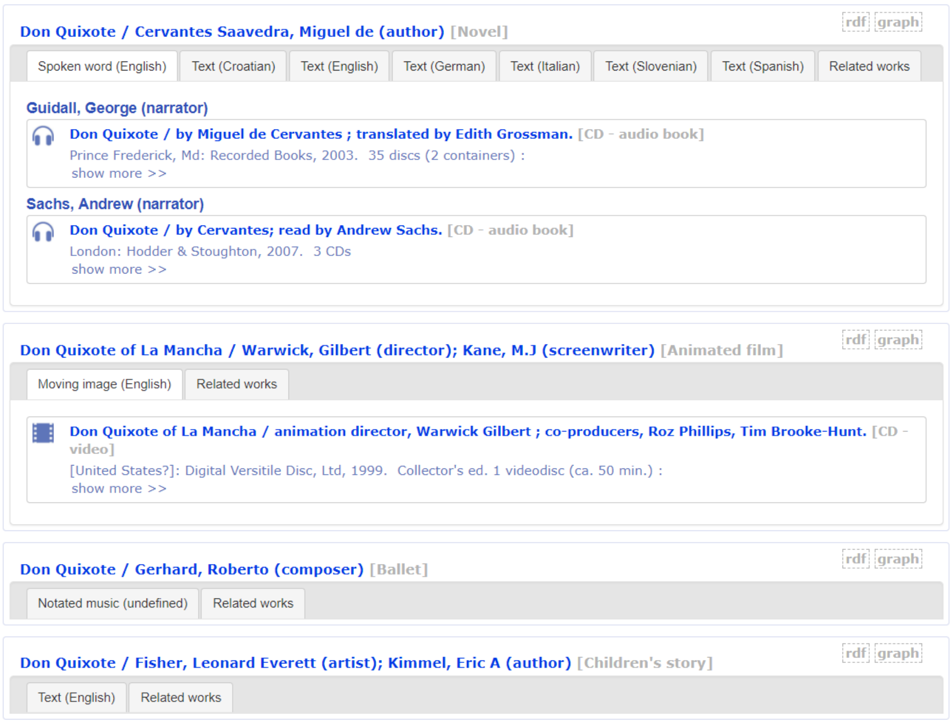
Match: All

Display: Manifestations

Ranking: Default

Select collection: Search all collections

Figure A1: Empty query in final UI with an overview of all the available facets


Don Quixote / Cervantes Saavedra, Miguel de (author) [Novel]

Guidall, George (narrator)

Don Quixote / by Miguel de Cervantes ; translated by Edith Grossman. [CD - audio book]
Prince Frederick, Md: Recorded Books, 2003. 35 discs (2 containers) :
show more >>

Sachs, Andrew (narrator)

Don Quixote / by Cervantes; read by Andrew Sachs. [CD - audio book]
London: Hodder & Stoughton, 2007. 3 CDs
show more >>

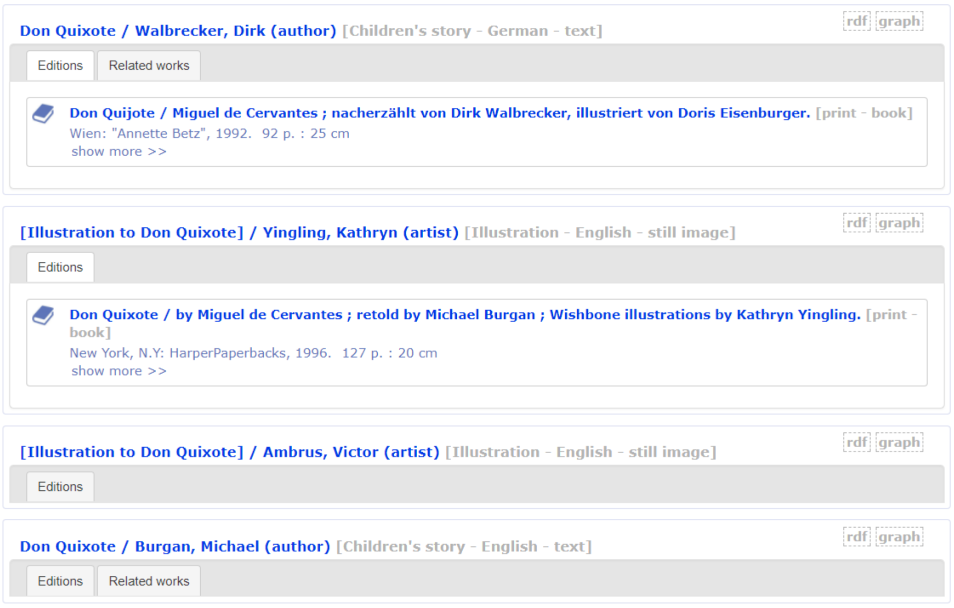
Don Quixote of La Mancha / Warwick, Gilbert (director); Kane, M.J (screenwriter) [Animated film]

Don Quixote of La Mancha / animation director, Warwick Gilbert ; co-producers, Roz Phillips, Tim Brooke-Hunt. [CD - video]
[United States?]: Digital Versitile Disc, Ltd, 1999. Collector's ed. 1 videodisc (ca. 50 min.) :
show more >>

Don Quixote / Gerhard, Roberto (composer) [Ballet]

Don Quixote / Fisher, Leonard Everett (artist); Kimmel, Eric A (author) [Children's story]

Figure A2: Work display


Don Quixote / Walbrecker, Dirk (author) [Children's story - German - text]

rdf graph

Editions | Related works

Don Quijote / Miguel de Cervantes ; nacherzählt von Dirk Walbrecker, illustriert von Doris Eisenburger. [print - book]
Wien: "Annette Betz", 1992. 92 p. : 25 cm
show more >>

[Illustration to Don Quixote] / Yingling, Kathryn (artist) [Illustration - English - still image]

rdf graph

Editions

Don Quixote / by Miguel de Cervantes ; retold by Michael Burgan ; Wishbone illustrations by Kathryn Yingling. [print - book]
New York, N.Y: HarperPaperbacks, 1996. 127 p. : 20 cm
show more >>

[Illustration to Don Quixote] / Ambrus, Victor (artist) [Illustration - English - still image]

rdf graph

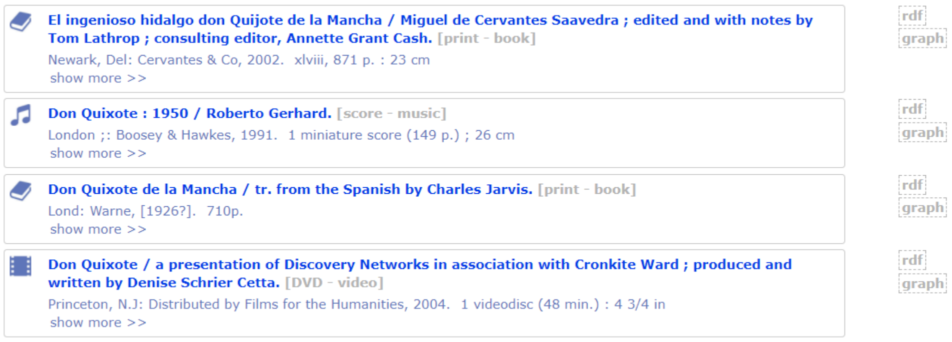
Editions

Don Quixote / Burgan, Michael (author) [Children's story - English - text]

rdf graph

Editions | Related works

Figure A3: Expression display


El ingenioso hidalgo don Quijote de la Mancha / Miguel de Cervantes Saavedra ; edited and with notes by Tom Lathrop ; consulting editor, Annette Grant Cash. [print - book]
Newark, Del: Cervantes & Co, 2002. xlviii, 871 p. : 23 cm
show more >>

rdf graph

Don Quixote : 1950 / Roberto Gerhard. [score - music]
London ;: Boosey & Hawkes, 1991. 1 miniature score (149 p.) ; 26 cm
show more >>

rdf graph

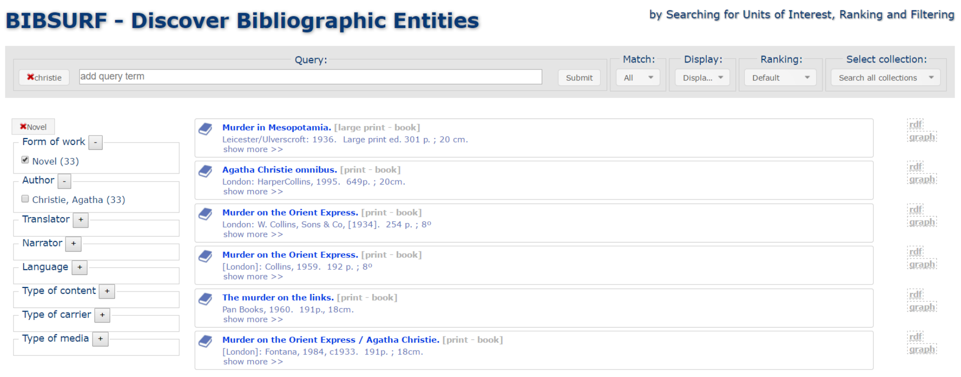
Don Quixote de la Mancha / tr. from the Spanish by Charles Jarvis. [print - book]
Lond: Warne, [1926?]. 710p.
show more >>

rdf graph

Don Quixote / a presentation of Discovery Networks in association with Cronkite Ward ; produced and written by Denise Schrier Cetta. [DVD - video]
Princeton, N.J: Distributed by Films for the Humanities, 2004. 1 videodisc (48 min.) : 4 3/4 in
show more >>

rdf graph

Figure A4: Manifestation display


BIBSURF - Discover Bibliographic Entities

by Searching for Units of Interest, Ranking and Filtering

Query: christie | add query term | Submit

Match: All

Display: Displa...

Ranking: Default

Select collection: Search all collections

Novel

Form of work -
- ☑ Novel (33)

Author -
- ☐ Christie, Agatha (33)

Translator +

Narrator +

Language +

Type of content +

Type of carrier +

Type of media +

Murder in Mesopotamia. [large print - book]
Leicester/Ulverscroft: 1936. Large print ed. 301 p. ; 20 cm.
show more >>

Agatha Christie omnibus. [print - book]
London: HarperCollins, 1995. 649p. ; 20cm.
show more >>

Murder on the Orient Express. [print - book]
London: W. Collins, Sons & Co, [1934]. 254 p. ; 8º
show more >>

Murder on the Orient Express. [print - book]
[London]: Collins, 1959. 192 p. ; 8º
show more >>

The murder on the links. [print - book]
Pan Books, 1960. 191p., 18cm.
show more >>

Murder on the Orient Express / Agatha Christie. [print - book]
[London]: Fontana, 1984, c1933. 191p. ; 18cm.
show more >>

rdf graph

Figure A5: BIBSURF UI original

Appendix B
Contract

I hereby with this contract agree that, Mei Jain Fung, can record the computer screen with audio included during my test. And permit the use of the test results to be included in the indicated masters's thesis at NTNU, during the academic year 2017.

I am informed that I will be kept anonymous in all published works.

Signature	Date