

Ilja Tuomas Salmio

# Library Management Applications

---

Helsinki Metropolia University of Applied Sciences

Bachelor of Engineering

Information and Communications Technology

Thesis

22.5.2014

Tekijä(t) Otsikko	Ilja Tuomas Salmio Library Management Applications
Sivumäärä Aika	34 sivua + 1 liite + sanasto 22. toukokuuta 2014
Tutkinto	insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Ohjelmistotekniikka
Ohjaaja(t)	Lehtori Vesa Ollikainen Professori Thomas A. Regelski
<p>Tässä insinööriyössä toteutetaan HomeLibrary -niminen MySQL-tietokantaa käyttävä kirjastonhallintaohjelmisto. HomeLibrary on helppokäyttöinen pienten yksityiskirjastojen hallintaohjelma. Insinööriyössä kuvataan ohjelmiston tausta ja tavoitteet sekä arvioidaan muiden kirjastonhallintaohjelmistojen soveltuvuutta tarkoitukseen.</p> <p>Tietokannan ja itse ohjelman rakenne näytetään, ja käyttöliittymä selitetään tyhjentävästi. Vaatimukset, jotka HomeLibrary tarvitsee toimiakseen tietokoneella, kerrotaan ja lopuksi projektin eteneminen selvitetään.</p>	
Avainsanat	ohjelmointi, tietokanta, Java, Hibernate, MySQL, kirjasto

Author(s) Title	Ilja Tuomas Salmio Library Management Applications
Number of Pages Date	34 pages + 1 appendix + Glossary 22 May 2014
Degree	Bachelor of Engineering
Degree Programme	Information and Communications Technology
Specialisation option	Software Engineering
Instructor(s)	Vesa Ollikainen, Senior Lecturer Professor Thomas A. Regelski
<p>In this thesis we implement HomeLibrary library management application that utilizes a MySQL database. The HomeLibrary is an easy to use library system meant for managing small private libraries by inexperienced users. The background information and reasons for production are explained as well as the requirements. Other solutions are discussed and their applicability to the problem is discussed.</p> <p>The structure of the database and the program is shown, and the user interface is explained in detail with examples. Finally everything that the HomeLibrary program needs from the computer in order to work is discussed, and how the project proceeded is covered.</p>	
Keywords	programming, database, Java, Hibernate, MySQL, library

## Glossary

1	Introduction	1
2	Goals and Background	2
3	Requirements for HomeLibrary Software	3
4	Freeware Alternatives	3
4.1	VuFind	4
4.2	Koha	5
4.3	Evergreen	6
4.4	Greenstone	7
4.5	OpenBiblio	8
4.6	PMB	9
4.7	Freeware Alternatives vs. Tailor-Made System	9
5	Design and Implementation	10
5.1	Overview	10
5.2	Database	11
5.3	Classes	15
5.4	User Interface	17
5.4.1	Navigation	17
5.4.2	Search View	18
5.4.3	Insert View	19
5.4.4	Edit View	20
5.4.5	New Book Advanced View	21
5.4.6	New Book View	23
5.4.7	New Book Quick View	23
5.4.8	New Person View	24
5.4.9	New Publisher View	26
5.4.10	New City View	27
5.4.11	New Country View	27
5.4.12	New Format View	28
5.4.13	New Language View	29
5.4.14	New Location View	29
5.4.15	New Role View	30
5.4.16	New Series View	30
5.4.17	New Tag View	31

6	System requirements and Performance	31
6.1	User Interfaces	32
6.2	Hardware Interfaces	32
6.3	Software Interfaces	32
6.4	Usage Intensity	32
6.5	Capacity Requirements	33
7	Discussion and Conclusions	33
	References	34
	Appendices	
	Appendix 1. The Example Book	

## Glossary

**Book** is a physical object in the library.

**CamelCase** (also spelled camel case or camel-case) or medial capitals is the practice of writing compound words or phrases in which the elements are joined without spaces, with each element's initial letter capitalized within the compound, and the first letter is either upper or lower case.

**CSS** (Cascading Style Sheets) shows how a particular document (like a website) looks. The font, textsize, background information etc. is written in CSS.

**Edition** is created when a newer version of the LanguageVersion has been printed.

**Foreign key** is a number that corresponds with the primary key of a row in another table.

**Hibernate** makes it possible for a java-program to communicate with a database.

**ILS** (Integrated Library System) is a professional library management system used to keep track of such things as what media items are in the library, which of these have been borrowed etc.

**Javascript** is a programming language often used in more complex websites.

**LanguageVersion** is created when the piece of work is translated into a new language (or first printing is made).

**Mapping-files** shows the program how the database is structured.

**MARC** (Machine-Readable Cataloguing) are a number of formats for the cataloguing of items in libraries.

**MARCXML** is a normal XML schema that conforms to the MARC21 standards.

**OPAC** (Online Public Access Catalog) is a library database that is used when the library's database has to be accessed from multiple computers. It is located in a server and is usually accessed via the internet.

**OpenSRF** (Open Scalable Request Framework) makes it easy for programmers without knowledge of the structure to create programs for Evergreen.

**Person** Many tables refer to the person-table. At the moment a person can be an author, translator or owner of the book. The nickname-table is also connected to the person-table. The person table has fields for: FirstName, LastName, DateOfBirth, Sex, Address, ZIPCode, Email and WWW. It has foreign keys called CountryID and NotesID.

**Piece of Work** is a kind of insubstantial idea of a media. It comprises all of the different forms of media in the world made about this (e.g. all of the books named Lord of the Rings in different languages, all of the dvd-movies, all of the bluray-movies, all of the audiobooks in different languages etc.) It has only its primary key, but many foreign keys from other tables reference it.

**POJO** has all of the constructors and the getters and setters in them. They help the program in communicating with the database.

**SDI** (Selective Dissemination of Information Service) is a tool meant to alert the user on the newest publications on a predetermined subject.

**SIP2** (Standard Interchange Protocol) is a communication protocol which is widely used for communication between library management systems.

**UNIMARC** is a MARC (see above) format used in Europe.

**XHTML** (Extensive HyperText Markup Language) is an XML version of the normal HTML language that the websites are written in.

**Z39.50** client-server protocol is used for searches from online databases.

## 1 Introduction

Most library management software in use can be hard to comprehend without necessary education. They are also mostly meant for big official libraries. HomeLibrary is a database software meant to give an easy to use platform for the management of different private media in one or more households for beginners.

Searching, adding and modifying information has been made easy, so the user doesn't have to be an expert in library administration.

In this thesis, Chapter 2 (entitled "Goals and Background") gives information on why the HomeLibrary was needed and what the purpose of this B. Sc. thesis is. It also has a biography on Prof. Thomas A. Regelski, for whom the program was made for.

Chapter 3 (entitled "Requirements for HomeLibrary Software") shows the requirements needed for the HomeLibrary. Some of the requirements were made by researching what a library management program needs in order to work. Other requirements were given by prof. Regelski (see Chapter 2) to make the program suitable for his own library.

Chapter 4 (entitled "Freeware Alternatives") provides short reviews on multiple open source Library management programs found on the internet. It also shows why they are not optimal for the purpose of the library in question.

Chapter 5 (entitled "Design and Implementation") is the biggest chapter in this thesis. It gives a comprehensive view on how the HomeLibrary program was designed and made. There are Sections on the database, classes and user interface. The Database Section shows the structure of the database and the purpose of each table in the database. The Classes subchapter deals with Section tells how the main program is structured and how the multiple classes communicate with each other. The subchapter on User Interface may be the easiest read for nonprogrammers. It has screenshots on the different windows in the program and has small tutorials on how each is used.



Chapter 6 (entitled “System Requirements and Performance”) shows what is needed in terms of both software and hardware for the HomeLibrary to work. It also offers tips on how to best use the program.

In chapter 7 (entitled “Discussion and Conclusions”) different problems that arose during the making of the program are considered together with how the final product compares to the goals and requirements given at the start of the project. Finally there are different ideas of how the HomeLibrary program can be further developed.

## **2 Goals and Background**

The purpose of this B.Sc. thesis project is to design and implement HomeLibrary software and to explore different library management systems freely available. This document covers the features and functionality for easy use of the HomeLibrary software. The software and hardware specifications as well as user interface description are also included.

The motivation for this project stems from the needs of Prof. Thomas A. Regelski who has collected a large number of books and important documents from around the world in which he has made notes in the marginalia. He would like to have a personalized management system for cataloguing them for his own use and the use of other academic researchers interested in his lifework.

Prof. Regelski is Distinguished Professor of Music (Emeritus), from State University of New York at Fredonia NY, where he taught choral conducting, secondary methods, and foundations courses in philosophy, psychology, and sociology. He has also been widely published in an array of journals in and outside of music education. He is the co-founder of the MayDay Group and editor of its e-journal, *Action, Criticism, and Theory for Music Education*, from its inception in 2002 until the summer of 2007. Author of *Principles and Problems of Music Education* (Prentice-Hall, 1975), *Arts Education and Brain Research* (Alliance for Arts Education/MENC, 1978), *Teaching General Music: Action Learning for Middle and Secondary Schools* (Schirmer Books, 1981), and *Teaching General Music in Grades 4-8: A Musicianship Approach* (Oxford University Press, 2004), and co-editor (with J.T. Gates) of *Music Education for Changing Times* (Springer, 2009), he is presently a Docent at Helsinki University (Finland), teaching

courses in research writing to graduate students in the Faculty of Behavioural Sciences, and he lectures occasionally in the music education department at the Sibelius Academy. [1]

### **3 Requirements for HomeLibrary Software**

The main requirements for HomeLibrary are the ease of use in both the finding of information and adding new information (new books, duplicates of existing books, persons, etc.) by inexperienced users. This is because Prof. Regelski is not a librarian.

Prof. Regelski asked that there would be a feature that informed him if a book had marginalia and index written by the owner.

Exporting lists of data to Excel was also a requirement, so the list could be easily inserted to other library management systems.

During the cataloguing of the library, time management became an issue. Since prof. Regelski lives in Helsinki, and the majority of his books are in upstate New York, cataloguing of the books can be done only during the summer months when prof. Regelski is visiting the United States of America. For this reason it would be faster to catalogue the books into an Excel-file and import the file to the database back in Finland, when the time allows.

### **4 Freeware Alternatives**

In this chapter many alternative open source library management systems found on the internet are reviewed and compared to the requirements set for the HomeLibrary. This is to give the user some options other than HomeLibrary as a platform for library systems.

## 4.1 VuFind

VuFind (Figure 1) [2] is an open source library management system made “by libraries, for libraries”. With VuFind the client can find and browse through the assets of the library by adding additional features to the standard OPAC. These features are: Catalogue Records, Digital Media, an Institutional Repository for saving and management of scientific documents and an Institutional Bibliography for the searching of the same and catalogues of other libraries.

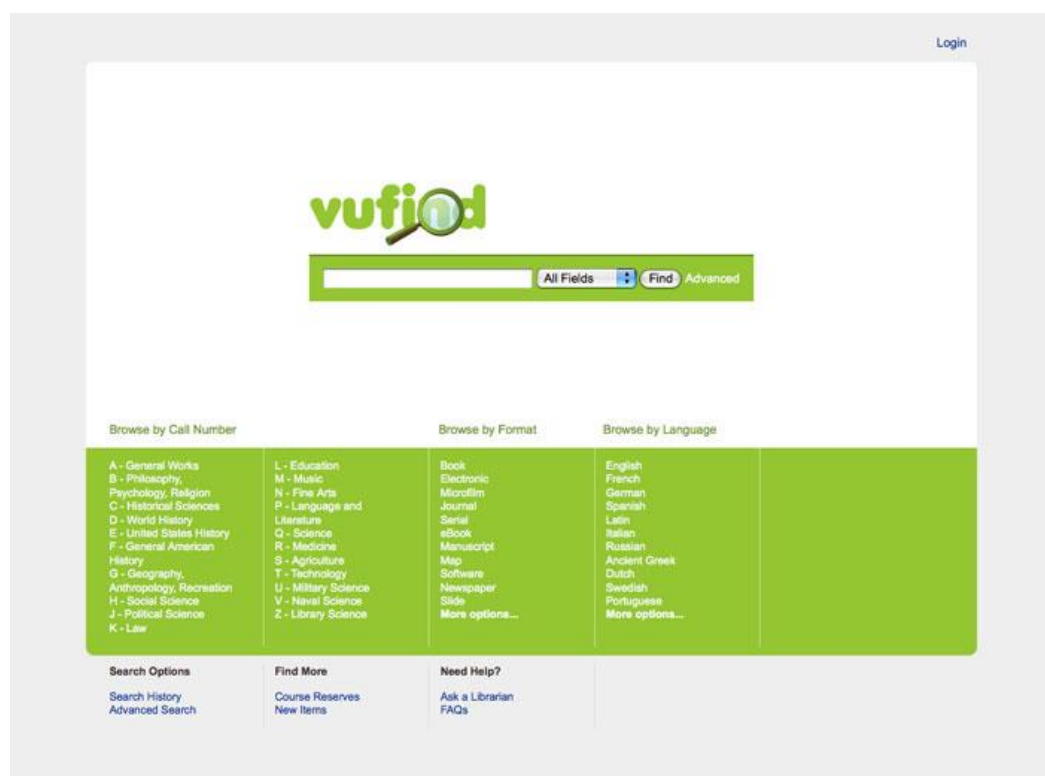


Figure 1. VuFind main page

Many of the features can be changed to suit the user’s needs and the open sourced nature of the code means it is constantly being added to and new features can be made to fit the needs of a particular library. [2]

While VuFind is very helpful while searching for information by library customers, the fact that it is so modular makes its installation a job for an IT-expert and adding new information a job for professional librarians. HomeLibrary on the other hand comes completely assembled, and is easy to manage by regular people without professional IT skills.

## 4.2 Koha

According to the developers' website "LibLime Koha (see Figure 2) [3], is an open source internet-based Integrated Library System (ILS), used world-wide by public, school and special libraries." [3]



Figure 2. Koha main page

Koha features a complex client management system including a way to add familial relationships (e.g. client is a parent of another client) and Clubs and Services feature (e.g. book clubs). It also offers a user friendly way of keeping tabs on items and a sophisticated way to manage reservations including making a reservation for a title or a specific item and a helpful way for the staff to manage reservation queues. If a book is old or obsolete, Koha offers a handy way to refresh it to a more modern edition. Koha has SIP2 communication protocol which is widely used for communicating with other library management systems. Koha is programmed for internet use with XHTML, CSS and Javascript. [3]

Koha can be intimidating for a beginner to use because of the overload of information in each window and like VuFind it is made for libraries not for home-use. Adding new information is also very time-consuming. There are also several costs that make it undesirable for home-use.

### 4.3 Evergreen

Evergreen (see Figure 3) is a widely used open source library management system. It manages both the functions meant for the public and the library's inner management (checkouts, checkins etc.). It also has the ability to share assets between different libraries. Evergreen is a web program and is based on OpenSRF framework. [4]

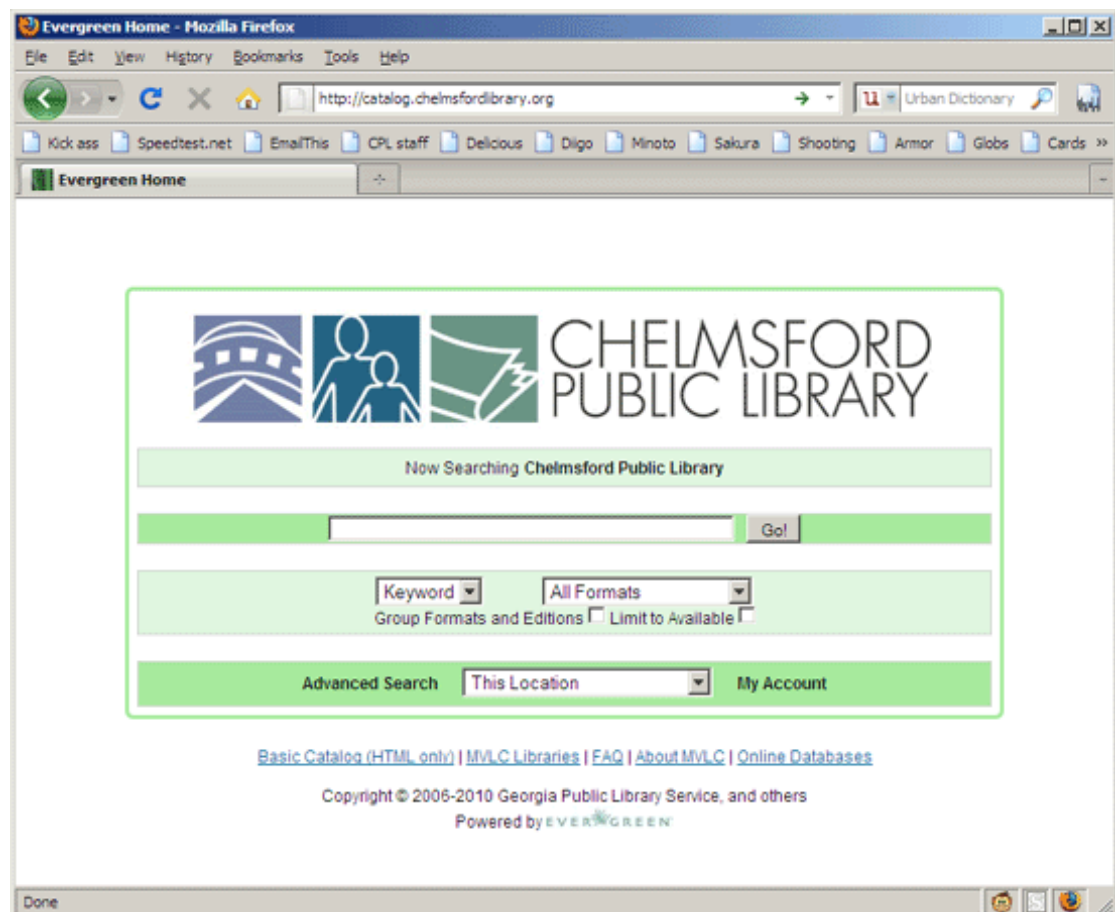


Figure 3. A library using Evergreen

Evergreen seems to be a good alternative for new users, but it has lots of publicly unknown technical terminology. There are also annual support costs and other costs that prohibit its use in private home libraries.

#### 4.4 Greenstone

The collection of open source applications known as Greenstone (see picture 4) is meant for managing and searching digital documents in a library. The formats it can handle include text, PDF, Word, html, jpg, tiff, MP3, video, etc. Most of the text-formatted documents are archived in XML-like Greenstone Archive Format (GAF). [5]

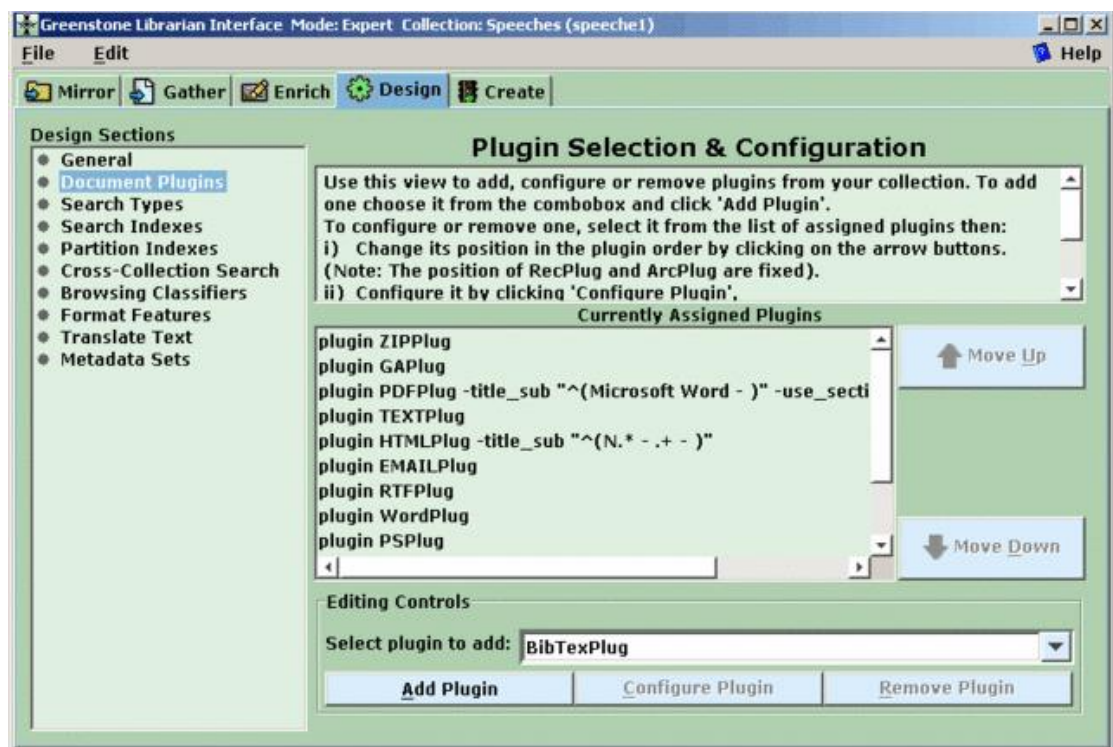


Figure 4. A screenshot of Greenstone

Since Greenstone is meant for storing only digital documents, it is unsuitable for the adding metadata of physical books.

## 4.5 OpenBiblio

OpenBiblio (see Figure 5) [6] is a multilanguage open source integrated library system (ILS) meant for small to medium-sized libraries. It has many easy-to-use features like check-in/out, client management, managing of the library's items (including MARC and MARCXML formats). The Online public access catalog (OPAC) of OpenBiblio is easy for patrons to use. Some of the more advanced features found in other systems are not found here. [6]

today's date: May 13, 2012  
library hours: M-F 8am-9pm, Sa noon-5pm, Su 1-5pm  
library phone: 111-222-3333

Home **Circulation** Cataloging Admin Reports

Logout

» Home  
[License](#)  
[Help](#)

Welcome to OpenBiblio

Use the navigation tabs at the top of each page to access the following library administration sections.





Tab	Description
 Circulation	Use this tab to manage your member records. <ul style="list-style-type: none"> <li>• Member administration (new, search, edit, delete)</li> <li>• Member bibliography checkout, holds, account, and history</li> <li>• Bibliography checkin and shelving cart list</li> </ul>
 Cataloging	Use this tab to manage your bibliography records. <ul style="list-style-type: none"> <li>• Bibliography administration (new, search, edit, delete)</li> </ul>
 Admin	Use this tab to manage staff and administrative records. <ul style="list-style-type: none"> <li>• Staff administration (new, edit, password, delete)</li> <li>• General library settings</li> <li>• Library collection list</li> <li>• Library material type list</li> <li>• Library theme editor</li> </ul>
 Reports	Use this tab to run reports on your library data. <ul style="list-style-type: none"> <li>• Report.</li> <li>• Labels.</li> </ul>

Figure 5. OpenBiblio main page

While OpenBiblio is the easiest to use by beginners, the installation is quite hard. Since it is a web-based program it needs a web server like Apache. This can be a problem to users with no knowledge of server upkeep. Since OpenBiblio is an ongoing project this may change.

#### 4.6 PMB

“PMB (PhpMyBibli) [7] is a fully featured open source integrated library system.” PMB features many of the normal library management system functions. There is the most important ability to checkin and –out of items and ability to see which items need to be updated. The ability to manage the metadata of the library items and the user management is also included. The user can get different Reports and PMB even has SDI (Selective Dissemination of Information Service) functionality (see Glossary).

PMB is very easy to use for both the staff and patrons. It uses the European UNIMARC standard. Z39.50 client-server protocol is used for searches. The database can be backed up. Adding and extracting of outside library records is also possible. PMB is still in development.

PMB is written in PHP programming language. [7]

Like OpenBiblio since PMB is a web-application it needs knowledge about the Apache web-server in order to install and maintain. PMB was also originally made in French and while it has an English version, their website (<http://www.sigb.net>) doesn't seem to have an English option.

#### 4.7 Freeware alternatives vs. tailor-made system

While the readymade applications are good, in that they are made by professionals. But they are also made for professionals. This is why they are not suitable for home use. Uneducated users can have a hard time doing different tasks and the applications can have too many features, making them too heavy for home computers. Also while the freeware are (as their name suggests) free, most still require initial investments (servers etc.) and/or upkeep costs.



## 5 Design and Implementation

In this chapter, we discuss the design and implementation issues of HomeLibrary system. There are also pictures of the different windows (called views) in the 5.4 Section along with explanations/tutorials on who they are used

In this chapter examples are used to show where the bibliographic data is stored and processed in the software. The example book chosen is “Java 2, Ohjelmoinnin peruskirja” (ISBN 8951-846-237-2). The pictures of the cover and the information page are shown in Appendix 1.

### 5.1 Overview

The database was designed first and the schema went through multiple revisions before it was programmed using MySQLWorkbench 5.2 CE. This was done because the database had to be as perfect as possible before coding. Trying to change the database in the middle of the development work would mean that the Java code would also have to be changed, which would cause unnecessary work. The programming of the main application was done with Java language using NetBeans IDE 5.3. This was because NetBeans is very MySQL-friendly, and there are many helpful NetBeans-tutorials found on the internet. The database was connected to the HomeLibrary program’s services.

A Hibernate configuration file (`hibernate.cfg.xml`) was made into the default package of the project tree. The configuration file has information on resource mappings, database connection etc. The `hibernate.cfg.xml` was modified in two ways. A true-valued `Hibernate.show_sql` and a `ClassicQueryTranslatorFactory`-valued `hibernate.query.factory_class` were added as Hibernate Properties. Next, a `HibernateUtil.java` file was created as a helper class (located in the `homelibrary.util` package). When the user clicks the `Homelibrary.jar` to start the program it is the `HibernateUtil.java` that loads the `hibernate.cfg.xml` and accesses `Hibernates SessionFactory` to obtain a `Session` object [3].

In order to reverse-engineer `Hibernate Mapping Files` and `Plain Old Java Objects (POJO)` from the database, a `hibernate.reveng.xml` reverse engineering file was made (lo-

cated in <default package>). All of the tables were specified in the hibernate reverse engineering wizard. As a result of the reverse-engineering process, the Mapping Files and POJOs are generated.

The basic design of the main program is ModelViewController-based (see Figure 6). First the MainView.java was created (see Sections 5.4.1- 5.4.4). Next the NewBookAdvancedView.java (5.4.5) was created and from that the NewBookView.java (5.4.6) and NewBookQuickView.java (5.4.7) were later added to make the adding of books easier. After that the smaller Views (5.4.8- 5.4.17) were added. After this was done the Controller.java and HomeLibraryModel.java were made and both were saved in their respective packages. The model contains all of the functions needed for the program, and the controller is responsible for passing information between the views and the model.

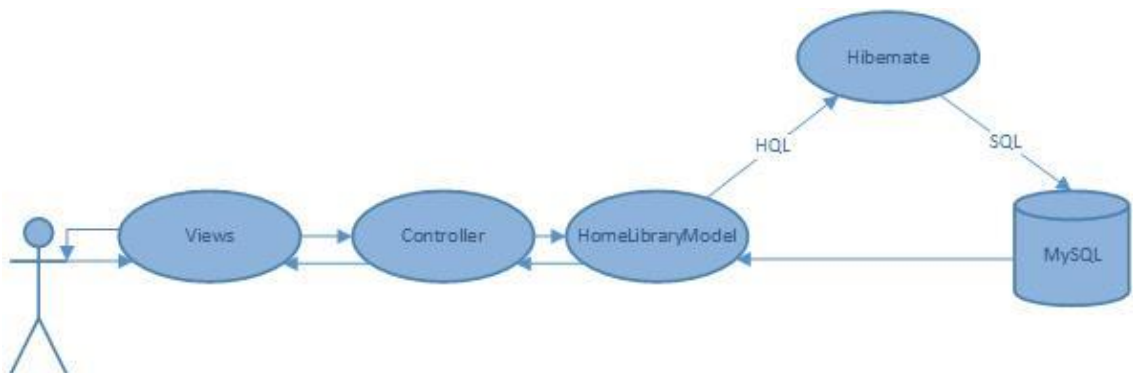


Figure 6. Program structure

## 5.2 Database

MySQL is one of the most used open source relational database management systems (RDBMS) in the world. It works on many different operating systems, which is why it was chosen as the RDBMS in this project.

Figure 7 shows the database structure. In the figure, table names are in lower case. Field names are in camel case (compound word that has the first letter of each word in capitals e.g. SubTitle, see Glossary) and the first letter is capitalized. The field name for foreign key is in most cases the same as the primary key of the table it refers to. The exceptions are: in 'book' table the "OwnerID" foreign key refers to 'person' table. In

'editions' table "PubCity" foreign key refers to 'city' table. In the 'languageversion' table the "Translator" foreign key refers to 'person' table.

In each table, ID is a unique integer (primary key) with auto-incrementation. ID is necessary for each table. ID is usually the same as the table name, except when it is also a foreign key.

There are many tables in the database, but the main tables are the books' "own" tables. The whole database is centred around these tables. They are connected to each other in a kind of tree-like relationship, where a parent can have many children, but a child can have only one parent. The table hierarchy goes (in a parent-to-child order) like this: 'pieceofwork'->'languageversion'->'edition'->'book'. The "granpa" of these tables is the 'pieceofwork' table (Instances of the 'pieceofwork' table are called from now on as 'pieceofworks'. Similar notations denoted by apostrophes, are used in instances of other tables). A 'pieceofwork' is an insubstantial idea of a media. It comprises all of the different forms of media in the world made about this (e.g. all of the books named Lord of the Rings in different languages, all of the DVD movies, all of the BluRay movies, all of the audiobooks in different languages etc.) It has only its primary key, but many foreign keys from other tables reference it. A foreign key is a number that corresponds with the primary key of a row in another table.

A 'pieceofwork' can have many 'languageversions'. A 'languageversion' is created when the piece of work is translated into a new language (or first printing is made). An instance of 'person' table is connected to 'languageversion' table by being the "translator" of the 'languageversion' (no one translated the example book). A 'languageversion' can have only one "translator". A 'languageversion' has also the "language" it was primarily written in (e.g. Finnish). This is why the 'languageversion' has a foreign key that refers to the 'language' table. The rest of the fields in 'languageversion' are "Title" (e.g. Java2)," SubTitle" (e.g. Ohjelmoinnin peruskirja) and "OriginalVersion" (the example book is the original, so there is no reason to insert anything into this field, but Java2 can be added) in which the original name of the first printing of the book can be added. If a new edition of the 'pieceofwork' is translated by someone else it is a new 'languageversion'.

There can be multiple printings of a 'languageversion'. This is why there is the edition-table. The 'edition' table has a "PubID" foreign key that connects it to the 'publisher'

table (e.g. Docendo Finland Oy). An 'edition' can have only one 'publisher'. If the 'edition' has more than one 'publisher', the 'publisher' is the publisher this 'edition' was printed by or the biggest named publisher or the first publisher in the list (if this 'edition' has many publishers). An 'edition' also has "FormatID" as a foreign key that connects it to the 'format' table (e.g. book). The "pubcityID" foreign key connects an 'edition' to the 'city' table by showing the city where the edition was printed (or made into its releasable form if the media is in non-printable format) (e.g. Jyväskylä). The Edition table also has fields for ISBN (or other pieceofwork identifier e.g. library of congress catalogue number, ISSN etc.) (e.g. 951-846-237-2), EditionNum for the number the edition is of the languageversion (e.g. 1) and the PubYear for the date the book was printed (e.g. 1 July 2005).

There can be multiple copies of the same edition in the library. The Book table refers to the physical books (or other media) in the library. The Book table has a foreign key to Person table called OwnerID. OwnerID shows the person who is the owner of the piece of the physical media (e.g. Ilja Salmio). PlacementID foreign key refers to the Placement table. Placement shows where in the library the book is located (e.g. office southwall). The Book table also has NotesID foreign key that refers to notes-table. Notes are any additional information that should be mentioned about the book (e.g. "This is the book used as an example."). The last foreign key in the Book table is the CountryBoughtID, which refers to the Country table by showing where the copy was bought (e.g. Finland). The data type of price field is varchar instead of int, so the user can put the name of the currency or the currency sign (\$, €, £ etc.) in with the price (e.g. 35 FIM). TimeAdded is in timestamp format and is automatically added when a new book is added to the library (e.g. 09.10.16.21.11.2013).

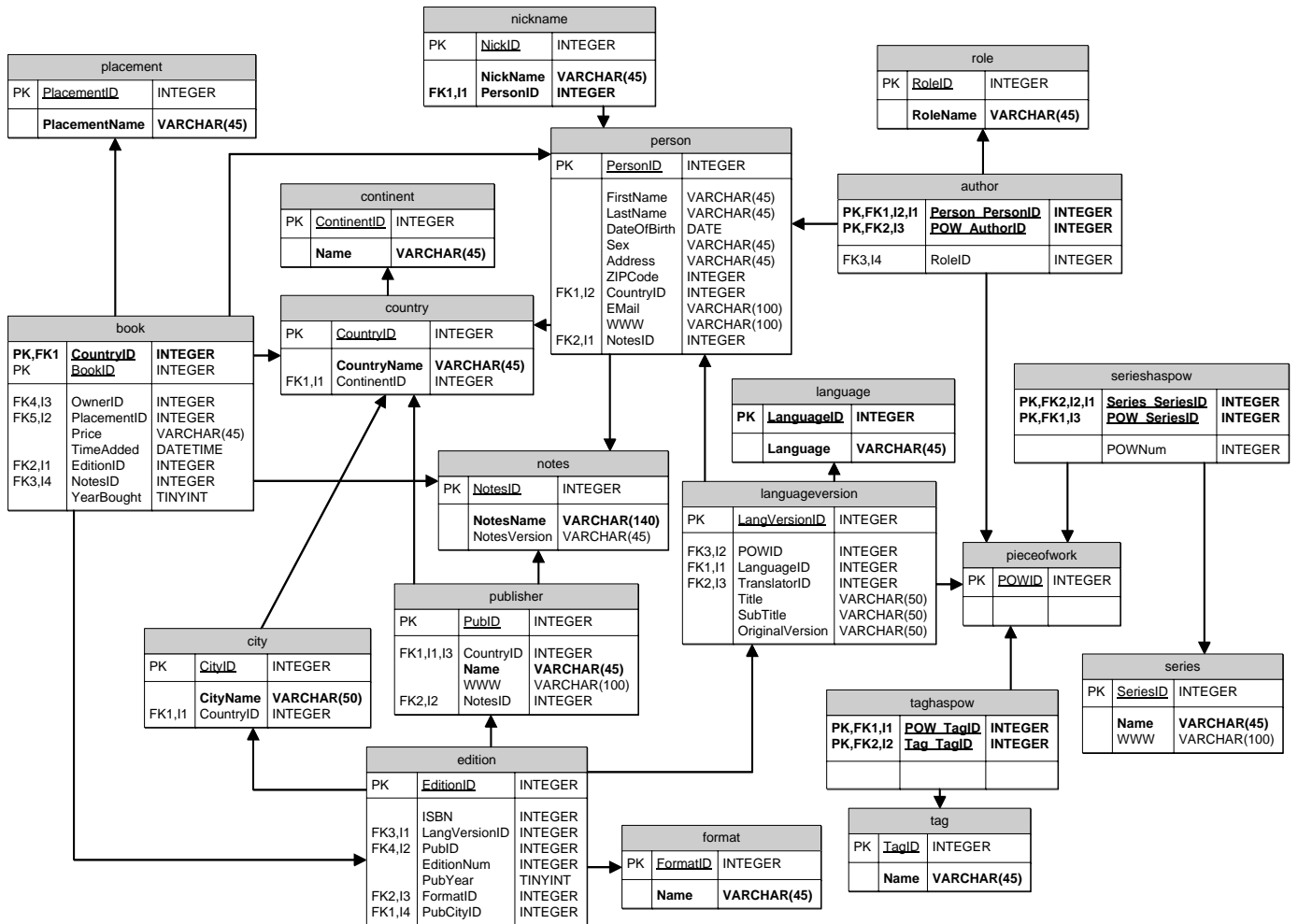


Figure 7. Database schema

An instance of person-table is connected to pieceofwork table by showing the author relationship between pieceofwork and person (a 'person' is an author of a 'pieceofwork'). A 'pieceofwork' may have multiple 'authors' (e.g. Pekka Kosonen, Juha Peltomaki, Simo Silander) and an 'author' may have many 'pieceofworks'. This is why the author table is necessary for the many-to-many relationship. Because a 'pieceofwork' can have many 'authors', the author table is connected to the role table to give more information on what capacity the 'person' involved in the making of the pieceofwork was 'pieceofwork' (e.g. writer) (At the moment there is no corresponding functionality in the Java implementation, but such functionality will be added as soon as possible).

A 'pieceofwork' can be part of a 'series' of 'pieceofworks' (e.g. peruskirjat kursseille ja itseopiskeluun, see book cover in Appendix 1). A 'pieceofwork' can be a part of many 'series' at the same time and a 'series' may have many 'pieceofworks', which is why serieshaspow table is needed for the many-to-many relationship. Serieshaspow table also has a POWNum-field to show which number of the 'series' the 'pieceofwork' is (this also is not fully implemented at the moment).

A 'pieceofwork' can also have multiple tags aka keywords (e.g. programming, Java, education) and a 'tag' can be in multiple 'pieceofworks'. This is why serieshaspow table is used to help with the many-to-many relationship.

Many tables refer to the person table. At the moment a 'person' can be an 'author', 'translator' or 'owner' of the 'book'. The Nickname table is also connected to the Person table. The Person table has fields for: FirstName (e.g. Ilja), LastName (e.g. Salmio), DateOfBirth (e.g. 1.3.1984), Sex (e.g. male), Address (e.g. Tiirismaantie 6 C 53), ZIP-Code (e.g. 00710), Email and WWW. It has foreign keys called CountryID (e.g. Finland) and NotesID (e.g. "The author of this program").

The Publisher table has fields for Name (e.g. Docendo Finland Oy) and WWW (<http://www.docendo.fi>). It also has foreign keys CountryID (e.g. Finland) and NotesID (e.g. "This publisher makes educational books").

### 5.3 Classes

In this section many of the most important java-classes are shown (see Figure 8) and the purposes of them are explained. The Class Diagram (see Figure 8) has all of the Views (see 5.4.1-5.4.17) as well as the POJOs (Plain Old Java Object) and the Model and Controller classes.

A POJO has all of the constructors and the getters and setters in them. They help the program in communicating with the database. The POJOs were created by reverse-engineering from the database, and each POJO has a corresponding database-table and an XML-mapping file with the same name. Some of the POJOs have been modified when needed.

The Model, the Views and the Controller were constructed from ground up. The user uses the View, which sends the data by Controller to the right part of the Model. The Model performs the necessary function and may return some data back to the Controller, which returns the data to the View.

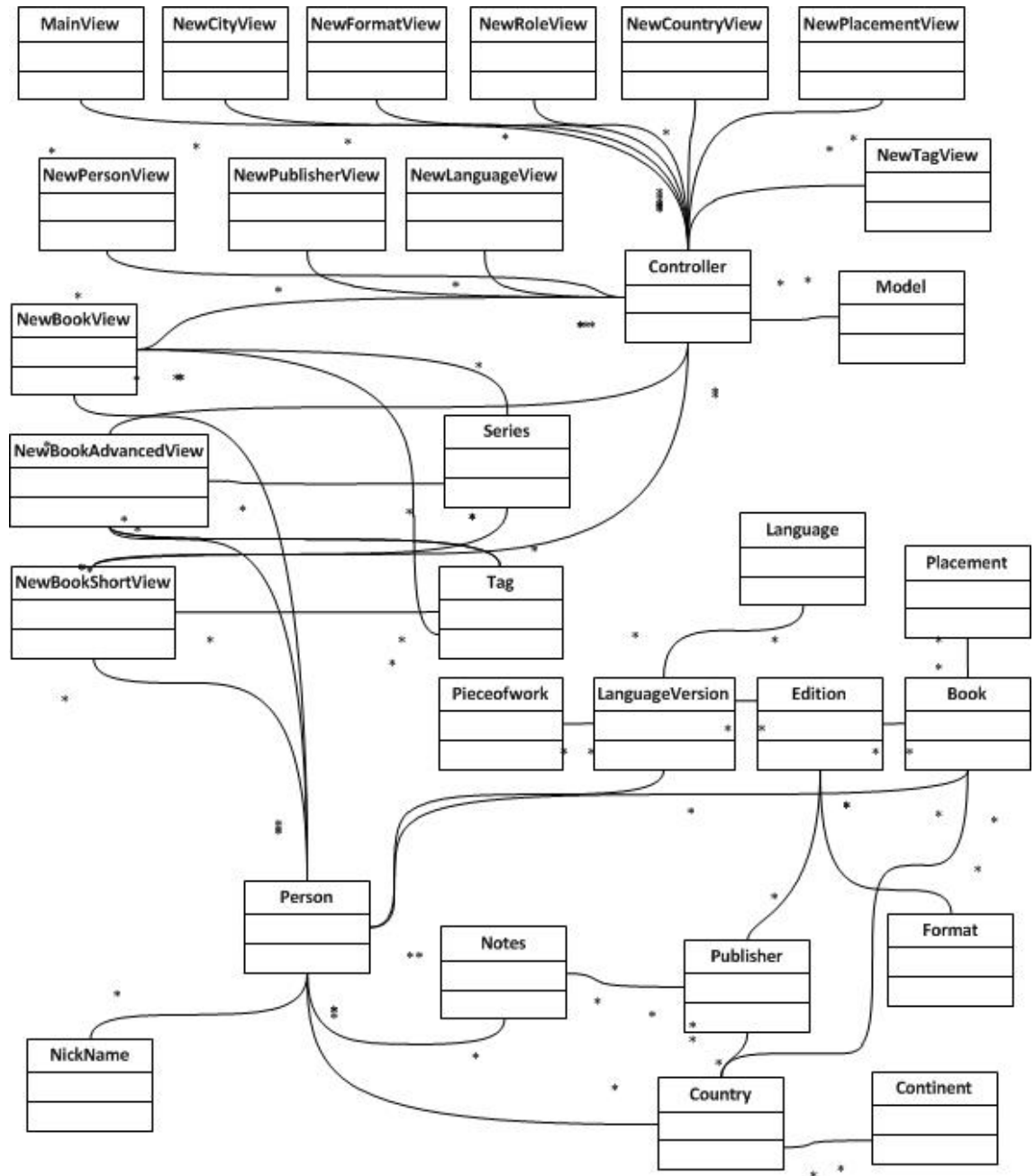


Figure 8. Class Diagram

## 5.4 User Interface

In this section what the user can see and do are shown and discussed. A View is a technical term that is a part of the ModelViewController software architecture pattern. For nonprogrammers a View can be thought of as a window within the program. A View tells the program how the information is presented to the user. The Views of the program are presented one by one in this chapter as well as the information on how to navigate between them. The program has a MainView, which has subViews in tabbed format. These subViews are SearchView, EditView and InsertView. All of the other Views are formed when the user clicks a button.

### 5.4.1 Navigation

The Navigation Diagram (see Figure 9) shows how the user navigates inside of the program. When the user (the stickfigure) starts the program, he/she sees the SearchView (5.4.2) inside the MainView. From here he/she can switch between the SearchView (5.4.2), the InsertView (5.4.3) and the EditView (5.4.4). From SearchView the user can open the three views related to a new book (NewBookView, NewBookAdvancedView, NewBookQuickView) (5.4.5 5.4.6 5.4.7). From InsertView the user can open all of the three new book views. From EditView the user can open the one of the specific edit views. Each of the three new book views has buttons connecting to the necessary other New\_\_\_View, so that the user doesn't have to go to the InsertView to find a way to add anything new.



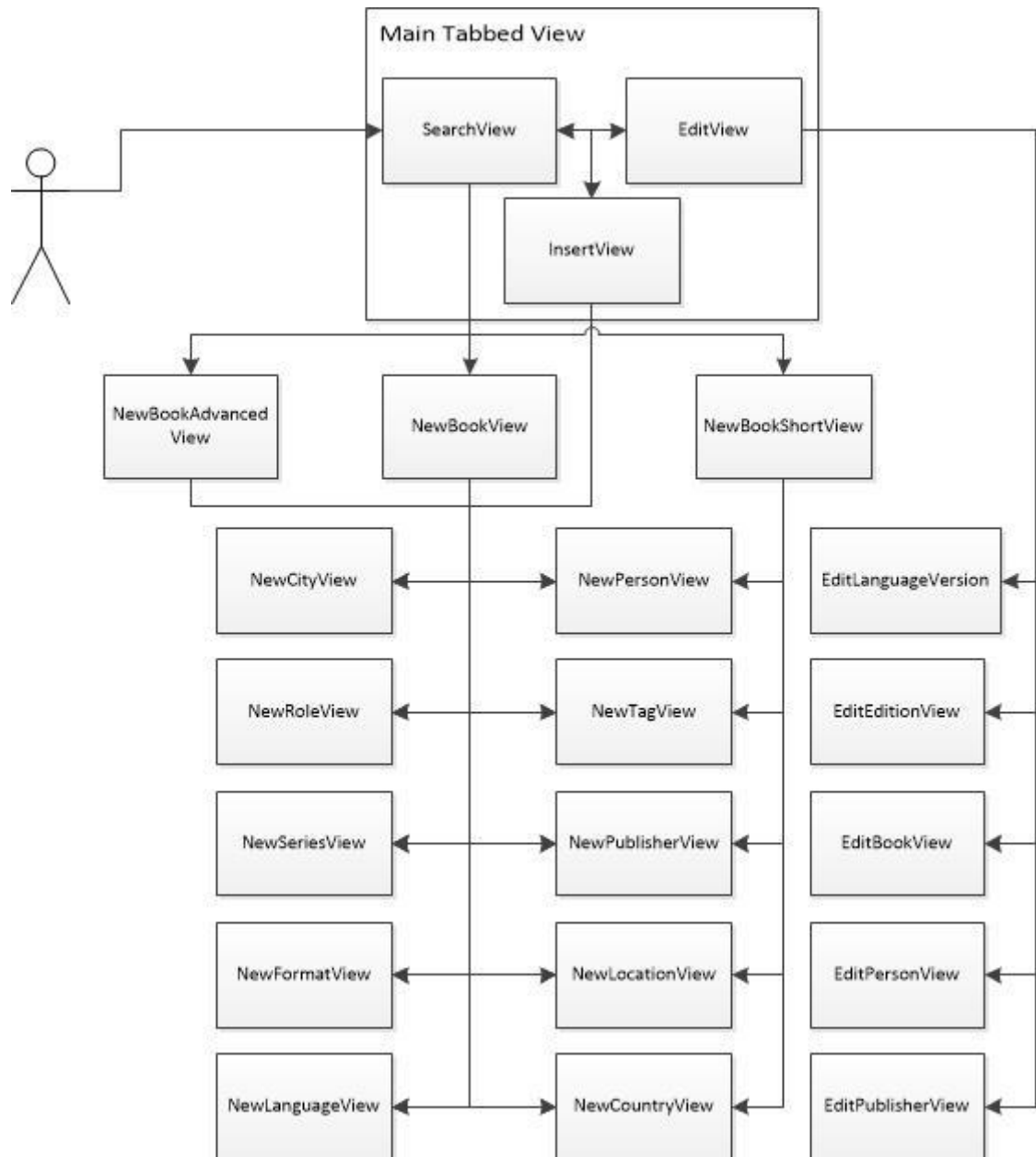


Figure 9. Navigation

#### 5.4.2 Search View

Search View is the primary view of the program. When the program is started, the first thing the user sees is the search view, and only from this window can the program be exited. Shutting the other windows of the program only shuts that window and not the whole program. Search View is a part of a group of tabbed views that can be seen on the top of the window.

In Search View (see Figure 10) the user can search books by title, author, language, translator, publisher, owner, tag, country, continent or location in the library. The user writes the search word into the searchTextField and selects the search parameter from the searchComboBox. Then he/she clicks the searchButton. The program searches the database and displays the results in the big searchTable (JTable) below. The order of the books is by BookID, but the order appearing in the searchTable can be changed by left clicking the top of the column the user wants to sort the list by (not yet fully implemented) and the order of columns can be changed by holding the left button down on top of the column and dragging the column to the preferred place in the list.

BookID	Title	Subtitle	Authors	Language	Original title	series	Translator	Publisher	Publication	Year bought	Country bo.	Time Added	Edition	ISBN	Publication	Format	Price	Location	Owner	Tags	First
1	ACT OF CR.	A study of t	Arthur. Ko	english		No Name	Dell Pupils.	New York NY	Other	2013-06-0	1	1	ACT OF CR.	1967-03-01	Book	-	Brocton NY	Thomas R.	Act Psychol.	First	
2	Education		Bertrand R.	english		No Name	Hercules Liv.	Unknown	Other	2013-06-0	1	1	097140010	1970-01-01	Book	-	Brocton NY	Thomas R.	Education		
3	The Gully.		Marshall, M.	english		No Name	The New A.	Toronto	Other	2013-06-0	1	1	-	1969-01-01	Book	0.63\$	Brocton NY	Thomas R.	Non-Fiction		
4	The Identity		J. Browns.	english		No Name	American	Unknown	Other	2013-06-0	1	1	0385001711	1971-01-01	Book	-	Brocton NY	Thomas R.	History Sci		
5	Beyond Fre.	A stunning	B. F. Siles	english		No Name	a Mentor B.	Unknown	Other	2013-06-0	5	5	Beyond Fre.	1961-01-01	Book	0.50\$	Brocton NY	Thomas R.	Bestseller		
6	The Age of	the 19th ce.	Henry D. A.	english		No Name	a Mentor B.	Brocton NY	Other	2013-06-0	5	5	The Age of	1961-11-01	Book	-	Brocton NY	Thomas R.	Philosophy		
7	The Age of	20th centur	Morton, W.	english		Mentor Boo	No Name	The New A.	Brocton NY	1961-11-01	Other	2013-06-0	7	The Age of	1961-11-01	Book	0.50\$	Brocton NY	Thomas R.	Philosophy	
8	Bertrand R.		Bertrand R.	english		Mentor Boo	No Name	The New A.	Unknown	1960-01-01	Other	2013-06-0	2	Bertrand R.	1960-01-01	Book	-	Brocton NY	Thomas R.	Psychology	
9	Age of Rea.	17th centur	Shuart, H.	english		Mentor Boo	No Name	The New A.	Unknown	1966-01-01	Other	2013-06-0	1	Age of Rea.	1966-01-01	Book	-	Brocton NY	Thomas R.	Science P.	
10	Introductio.		Leonard P.	english		Mentor Boo	No Name	The New A.	New York NY	1967-01-01	Other	2013-06-0	9	Introductio.	1967-01-01	Book	-	Brocton NY	Thomas R.	Philosophy	
11	The Virtue		Am. Rand	english		No Name	The New A.	New York NY	1964-01-01	Other	2013-06-0	22	0451005497	1964-01-01	Book	1.74\$	Brocton NY	Thomas R.	Philosophy		
12	The Spirit o.	A brilliant a	Kurt, Adam	english		No Name	Dom Justin	Doubleday	Garden Ct.	1962-01-01	Other	2013-06-0	11	The Spirit o	1962-01-01	Book	-	Brocton NY	Thomas R.	Religion, M.	
13	The Peter	Why things	Raymond.	english		No Name	WISQY	Brocton NY	1969-01-01	Other	2013-06-0	13	553054331	1969-10-01	Book	-	Brocton NY	Thomas R.	Bestseller		
14	Understan.	A popular n	James K.	english		No Name	Dell Pupils.	New York NY	1975-01-01	Other	2013-06-0	1	440082181	1975-01-01	Book	-	Brocton NY	Thomas R.	Marginalia		
15	the creative		Brewster.	english		Mentor Boo	No Name	The New A.	New York NY	1962-01-01	Other	2013-06-0	1	the creative	1962-01-01	Book	-	Brocton NY	Thomas R.	Marginalia	
16	The ABC of		Bertrand, R.	english		Mentor Boo	No Name	The New A.	New York NY	1969-01-01	Other	2013-06-0	18	0451616480	1969-01-01	Book	-	Brocton NY	Thomas R.	Philosophy	
17	Noam Cho.		John, Lyons	english		Modem Ma	No Name	The Viking	New York NY	1977-01-01	Other	2013-06-0	9	070-01911-9	1977-01-01	Book	2.45\$	Brocton NY	Thomas R.	Marginalia	
18	IrrationalMan	A Study in	William Sa	english		No Name	Doubleday	Unknown	1962-01-01	Other	2013-06-0	9	IrrationalMan	1962-01-01	Book	2.95\$	Brocton NY	Thomas R.	Philosophy		
19	The Huma.	Our Everg	Ashley, Mo.	english		No Name	Grove Pres.	New York NY	1964-01-01	Other	2013-06-0	1	The Huma	1964-01-01	Book	1.75\$	Brocton NY	Thomas R.	Philosophy		
20	Marx	and freedom	Terry, Eagl	english		The Great	No Name	Phoenix	London	1998-01-01	Other	2013-06-0	5	0 753 8918	1998-01-01	Book	-	Brocton NY	Thomas R.	Marginalia	
21	The Last d	Euthyphro i	Plato.	english		No Name	Penguin Cl.	Hugh Tred	London	1991-01-01	Other	2013-06-0	1987	0 14 04 40	1987-01-01	Book	1.98\$	Brocton NY	Thomas R.	Non-Fiction	
22	The two cul.		C. P. Snow	english		Mentor Boo	No Name	The New A.	Cambridge	1963-01-01	Other	2013-06-0	4	The two cul	1963-01-01	Book	-	Brocton NY	Thomas R.	Science M.	
23	The Lives o	Notes of a	Lewis, Tho.	english		No Name	Bantam/Vik.	Unknown	1975-01-01	Other	2013-06-0	7	0-453-101	1975-01-01	Book	1.95\$	Brocton NY	Thomas R.	Science M.		
24	Exploring t	Spill minds	Joseph Chu	english		No Name	Pocket Book	Unknown	Other	2013-06-0	0	0	071-89129-5	1975-01-01	Book	1.75\$	Brocton NY	Thomas R.	Non-Fiction		
25	The HomeI	Modernizat	Peter Berg	english		No Name	WISQY	Brocton NY	1974-01-01	Other	2013-06-0	0	0-394-719	1974-01-01	Book	2.95\$	Brocton NY	Thomas R.	Psychology		
26	The Tippin	How little B	Malcom, Cl.	english		No Name	Lillie Brow.	Unknown	2001-01-01	Finland	2013-06-1	1	0-316-679	2001-05-01	Book	76mk	Brocton NY	Thomas R.	Bestseller		
27	Stalin's th	on the mac	Lebok, Bevt	english		No Name	Bantam Sp.	Unknown	1978-01-01	Other	2013-06-1	4	0-453-123	1978-03-01	Book	-	Brocton NY	Thomas R.	Psychology		

Figure 10. Search View

### 5.4.3 Insert View

The Insert View (see Figure 11) is the view where the user can go to any of the “Add new” views. The user only has to press a button to go to the new view. All of the buttons can be found in context on other views. Insert View is a part of a group of tabbed views that can be seen on the top of the window.

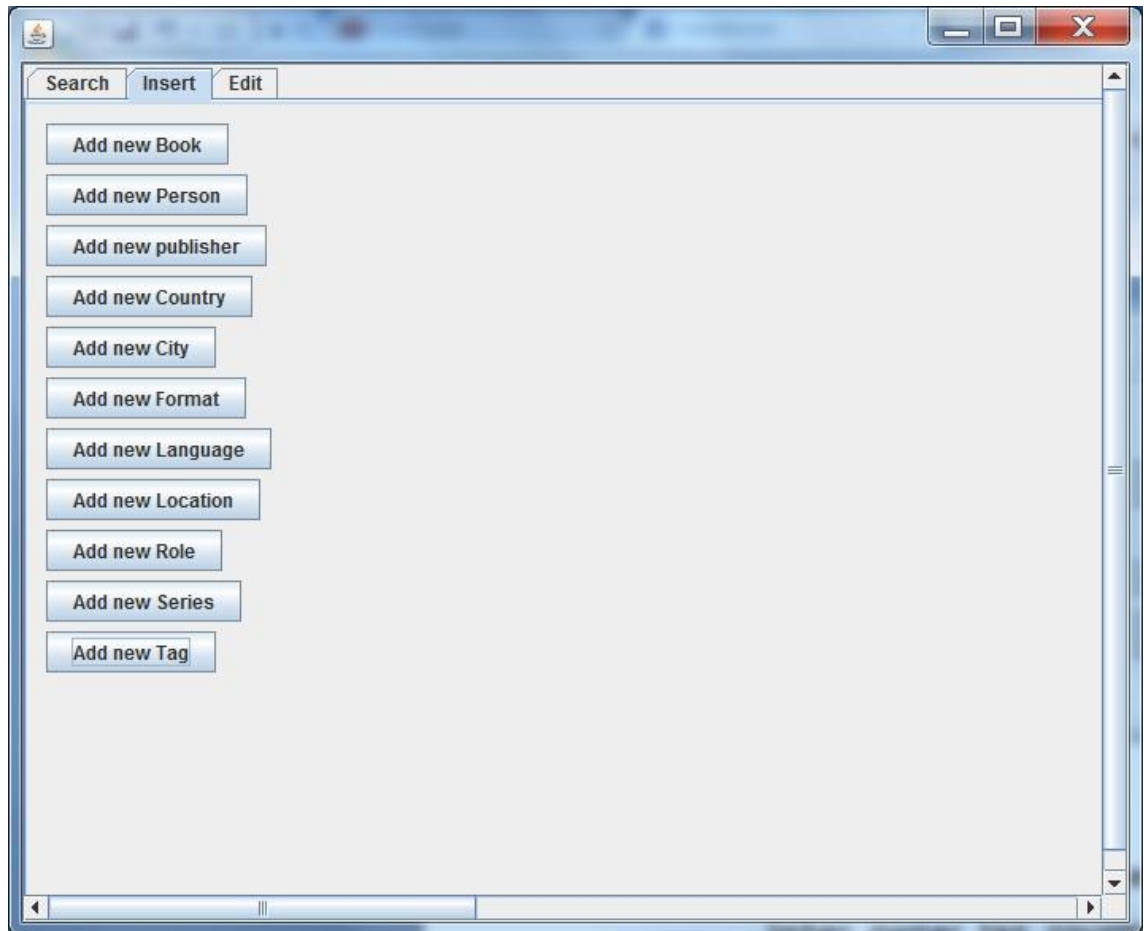


Figure 11. Insert View

#### 5.4.4 Edit View

If information in the database is wrong, the user can go to change it from the Edit View (see Figure 12). At the moment this is still a work-in-progress and not functional. The user selects the data he/she wants to change and the program opens one of the specific edit views with the textfields etc. already filled with the data.

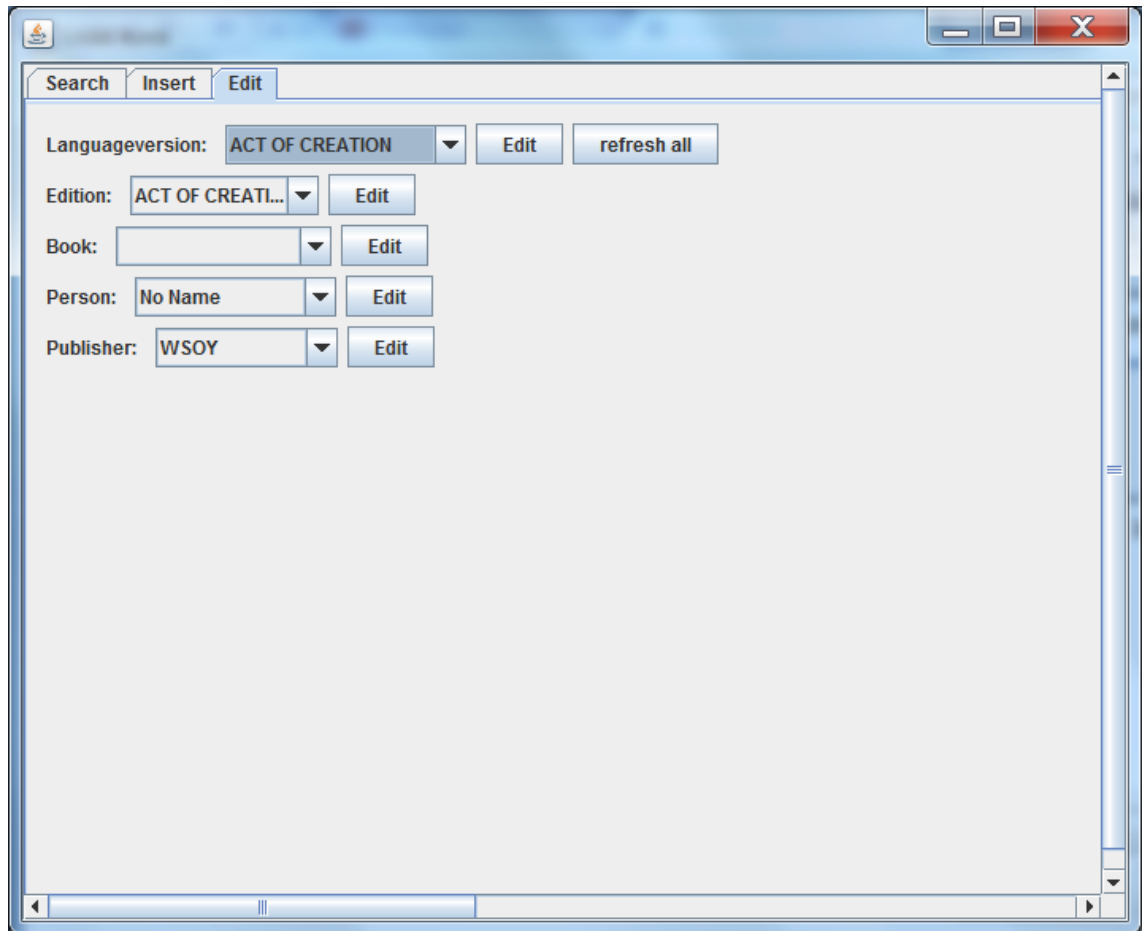


Figure 12. Edit View

#### 5.4.5 New Book Advanced View

The NewBookAdvancedView (see Figure 13) is used to add new rows to the pieceofwork, languageversion, edition and book tables. The NewBookAdvancedView is the most complex View in the program. It is used if a library has multiple instances of the same book. The user can input a new book to an already existing edition or a new edition to an existing LanguageVersion.

The screenshot shows a window titled "newBookFrame" with a light blue header. The main area is divided into several sections:

- Piece of work:** Contains a "Refresh All" button, an "Add Tag" section with a dropdown (value: "Art"), a "to list->" button, and a "Delete Tag from the list" button. It also has "Add new Tag to DB" and "Add new Role to DB" buttons.
- Add Author:** Includes a dropdown (value: "No Name"), an "as:" dropdown (value: "Writer"), a "to list of Authors->" button, and a "Delete Author from the list" button. It also has "Add new Person to DB" buttons.
- Piece of work is part:** Features a dropdown (value: "SeriesTest"), a "to list of series->" button, and a "Delete Series from the list" button. It also has "Add new series to DB" buttons.
- Languageversion:** Includes fields for "Title", "Subtitle", and "Original Title". It also has "Translator" (dropdown: "No Name"), "Language" (dropdown: "english"), and "Add new Language to DB" buttons. A "Add Languageversion to DB" button is at the bottom.
- Edition:** Includes "Edition is part of Languageversion:" (dropdown: "ACT OF CREATION"), "Edition Number" (text field), "Publisher" (dropdown: "WSOY"), "Publication Date dd-mm-yyyy:" (text field), "Publication City (unknown if no mention):" (dropdown: "Unknown"), "ISBN (Title if no isbn):" (text field), and "Format" (dropdown: "Book"). It has "Add new Publisher to DB", "Add new City", and "Add new Format" buttons, and an "Add Edition to DB" button at the bottom.
- Book:** Includes "Book is of Edition:" (dropdown: "ACT OF CREATION"), "Number of Pages" (text field), "Year Bought yyyy:" (text field), "Price" (text field), "Owner" (dropdown: "No Name"), "Country Bought ('other' if no mention):" (dropdown: "Other"), and "Location in the Library:" (dropdown: "Brocton NY USA"). It has "Add new Person to DB", "Add new Country to DB", and "Add new Location to DB" buttons, and an "Add Book" button at the bottom.
- Additional Information:** A large empty text area for notes.

Figure 13. NewBookAdvancedView

The most complex functions in this window are the Add Tag, Add Author and Add Series functions. They all work in a similar way. The left side combobox in the row shows all of the Tags/Authors/Series in the database. The right side combobox shows all of the Tags/Authors/Series that the user has chosen to be added to the piece of work. First the user chooses which Tag/Author/Series is to be added to the right side combobox from the left side combobox. Then he/she pushes the "to list->" button in the same row. The program adds the Tag/Author/Series to the right side combobox that shows the list of Tags/Authors/Series about to be added to the piece of work (these steps can be done as many times as wanted to add items to the list). Next to the list is the Delete Tag/Author/Series from the list in which the user can undo any Tag/Author/Series from the right side combobox before the list is inserted to the database by clicking the "Add LanguageVersion to DB" button.

### 5.4.6 New Book View

The NewBookView (see Figure 14) is used to add new rows to the pieceofwork, languageversion, edition and book tables. The NewBookView is a more simplified version of the NewBookAdvancedView. This is used when a completely new book (that doesn't have any other versions in the library) is added to the library. Only the "Add Book" is needed to save the information into the database.

The screenshot shows a web application window titled "New Book". The form is organized into four main sections, each with a "Refresh" button in the top right corner:

- Piece of work:** Contains dropdown menus for "Add Tag" (set to "Art"), "Add Author" (set to "No Name"), and "Piece of work is part of series" (set to "SeriesTest"). Each dropdown has a "to list->" button and a "Delete [X] from the list" button. There are also "Add new Tag to DB", "Add new Person to DB", and "Add new series to DB" buttons.
- Languageversion:** Includes text input fields for "Title", "Subtitle", and "Original Title". It also has dropdown menus for "Translator" (set to "No Name") and "Language" (set to "english"). Buttons for "Add new Person to DB" and "Add new Language to DB" are present.
- Edition:** Features input fields for "Edition Number (0)", "Publication Date dd-mm-yyyy", and "ISBN (Title if no isbn)". Dropdown menus include "Publisher" (set to "WSOY"), "Publication City (unknown if no mention)" (set to "Unknown"), and "Format" (set to "Book"). Buttons for "Add new Publisher to DB", "Add new City", and "Add new Format" are included.
- Book:** Contains input fields for "Number of Pages", "Year Bought yyyy", and "Price". Dropdown menus for "Owner" (set to "No Name"), "Country Bought ('other' if no mention)" (set to "Other"), and "Location in the Library" (set to "Brocton NY USA") are shown. Buttons for "Add new Person to DB", "Add new Country to DB", and "Add new Location to DB" are provided.

At the bottom of the form is a large text area for "Additional Information" and an "Add Book" button.

Figure 14. NewBookView

### 5.4.7 New Book Quick View

The NewBookQuickView (see Figure 15) is used to add new rows to the pieceofwork, languageversion, edition and book tables. The NewBookQuickView is meant for times

when books have to be added in a hurry. Only the bare minimum of information is added in order to identify the book. If more information is needed it can be found on the internet.

Figure 15. NewBookQuickView

The user first adds the authors, tags and series from the available choices to their respective lists which will eventually go to the book's information. Then he/she inserts the title and subtitle of the book. Then the publisher is chosen from the list and the identification code is inserted. Finally the owner, Country Bought and Location are selected and any additional information is inserted and the Add Book button is pressed.

#### 5.4.8 New Person View

The NewPersonView (see Figure 16) is used to add new rows to the Person table. The NewPersonView is used if the user can't find the name of the Author/Translator/Owner

from the database. It also has a table (implemented by Java jTable class) of the persons in the database. Whenever there is a list of users in the HomeLibrary program, there usually is a button next to it that, when pushed, directs the user here.

The user can insert the First name, Last name, NickName, Date of Birth, Address, ZIP code, Email, WWW-website and Additional Information. He/she can also choose the sex of the person and the home country. Once all of the known information is made the user clicks the Add-button to add the information to the database.

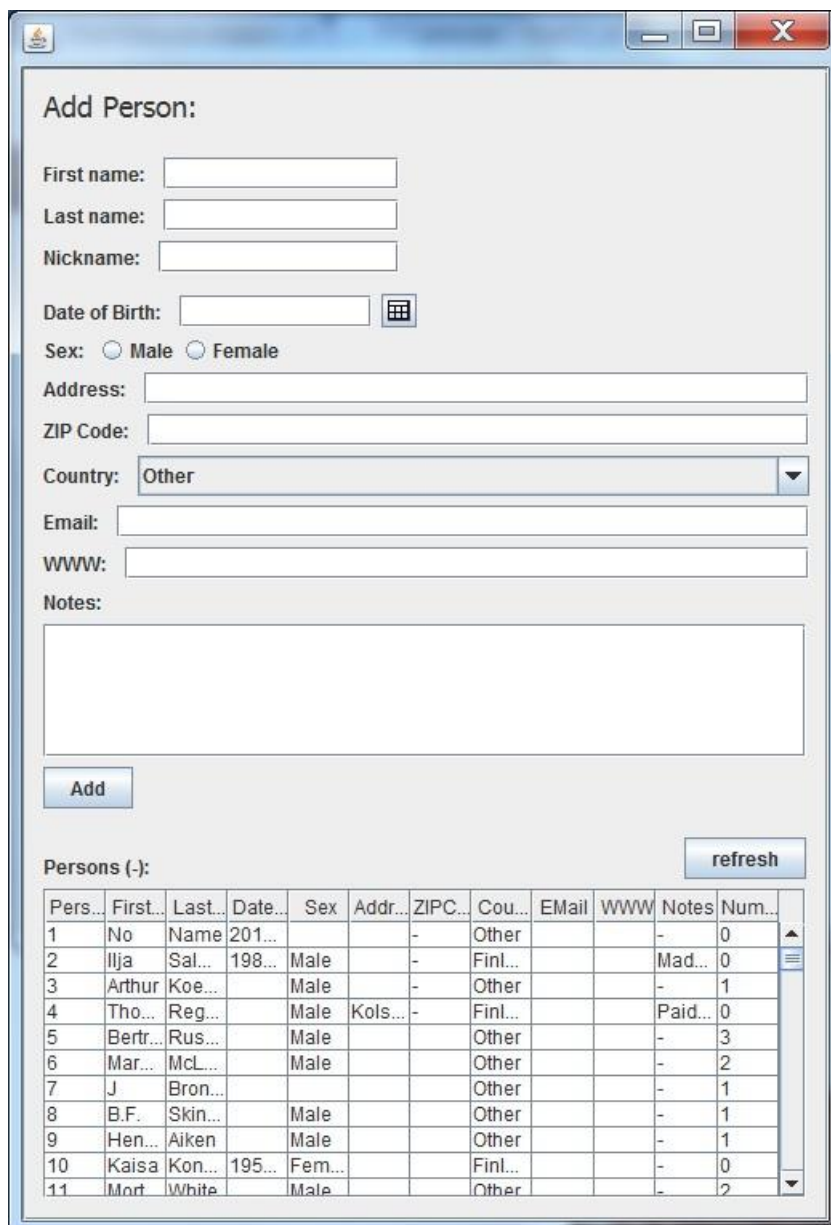


Figure 16. NewPersonView



### 5.4.9 New Publisher View

The NewPublisherView (see Figure 17) is used to add new rows to the publisher table. The NewPublisherView is meant for adding new publishers into the database. In the Name textfield the user inserts the name of the publisher (e.g. Docendo). The country dropdown (e.g. Finland) is meant for the country where the main office of the publisher is located. The WWW-textbox is for the website of the publisher (e.g. www.docendo.com). Additional Information goes to the Notes table in the database (e.g. schoolbooks). After all of the necessary information has been inserted, the user clicks on the “Add publisher” button to add the information to the database. On the bottom there is a list of all of the publishers already in the database along with the Number of media by the publisher.

Publish...	Country	Name	WWW	Notes	Num. of ...
1	Finland	WSOY		-	8
2	United ...	Dell Pu...		-	13
3	United ...	Horace ...		-	1
4	United ...	The Ne...			44
5	United ...	America...			1
6	United ...	Bantam ...			25
7	United ...	a Mento...			3
8	United ...	Double...			12
9	United ...	The Viki...			2
10	United ...	Grove P...			7
11	United ...	Phoenix			1

Figure 17. NewPublisherView

#### 5.4.10 New City View

The NewCityView (see Figure 18) is used to add new rows to the City table. In NewCityView the user can add new cities to the database. First the country that the city is located in is selected from the “Country” combobox (e.g. Finland). If the country cannot be found, a new country can be added by clicking the “new country” button, which opens the NewCountryView (5.4.11). Then the user adds the name of the city (and province/state/etc.) to the “City Name” textfield (e.g. New York NY). Finally the “Add City” button is pressed, which adds the city to the database. The successful input adds the city’s name to the combobox at the bottom of the window.

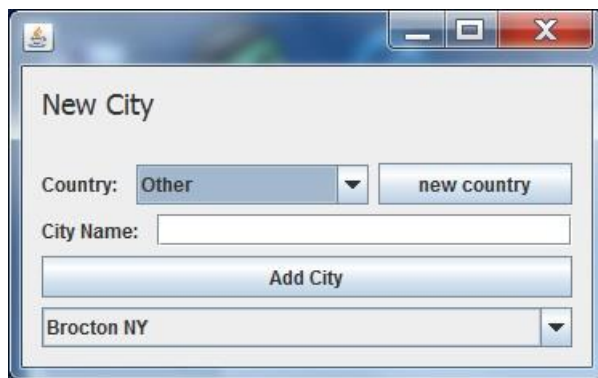


Figure 18. NewCityView

#### 5.4.11 New Country View

In NewCountryView (see Figure 19) new countries can be added to the Country table in the database. Most of the countries have already been added to the database, but any changes in the geopolitical situation can change the name of a country or there can be old countries that don't exist anymore (Soviet Union, Czechoslovakia, etc.).

The screenshot shows a window titled "New Country View". At the top, there is a text input field labeled "Add Country (No commas):". Below it is a dropdown menu labeled "Continent:" with "Other" selected. An "Add" button is positioned below the dropdown. Underneath is a table with the title "Countries" and a "Refresh" button to its right. The table has three columns: "CountryID", "Country", and "Continent". The table is currently empty.

CountryID	Country	Continent

Figure 19. New Country View

First the name of the new country is added to the “Add Country” textfield without any commas (e.g. Finland). Then the continent is selected from the “Continent” combobox. More continents can’t be added to the database since all of the continents have been discovered. Finally the “Add”-button is pushed which inserts the country to the database and to the “Countries” jTable.

#### 5.4.12 New Format View

The NewFormatView (see Figure 20) is meant for inserting new media formats (book, cd-rom, dvd, pdf, etc.) to the format table in the database. First the format is named by inputting the name to the “Name of the format” textfield (e.g. Book). Then the “Add format” button is pressed, which adds the format to the database and to the combobox in the bottom of the window.



Figure 20. NewFormatView

#### 5.4.13 New Language View

In the NewLanguageView (see Figure 21) new languages can be added to the Language table in the database. The name of the language is added to the “Name of the Language” textfield (e.g. English) and the “Add Language” is pressed in order to add the language to the database.



Figure 21. NewLanguageView

#### 5.4.14 New Location View

The location or placement shows where the book is located in the library. The location can be a house, a room, a cabinet or a shelf designated by the user. In NewLocationView (see Figure 22) a new location can be added to the Placement table in the database. The name of the location is inserted into the “Name of the location”-textfield (e.g. libraryroom southwall) and the “Add location” button is pressed. This adds the location to the database and to the combobox in the bottom of the window.

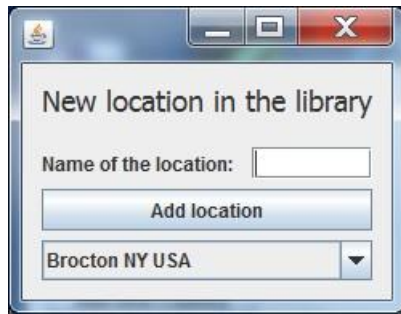


Figure 22. NewLocationView

#### 5.4.15 New Role View

In NewRoleView (see Figure 23) a new role for what an author did for the media can be made to the Role table in the database. In order to add a new role to the database the user must give a role name to the “Role Name” textfield (e.g. writer) and click the “Add role” button. The user can see which roles have been added to the database by looking at the combobox at the bottom of the window.



Figure 23. NewRoleView

#### 5.4.16 New Series View

If a media is a part of a series, then the series has to be on the Country table in the database. For this reason the NewSeriesView (see Figure 24) was created. The user needs to add the name of the series to the “Name of the series” textfield (e.g. peruskirjat) and click the “Add series” button so the series is added to the database.



Figure 24. NewSeriesView

#### 5.4.17 New Tag View

Tags are keywords that help find the media. The media can have multiple tags. Tags can be for example a genre. A NewTagView (see Figure25) works just like the previous Views. It is used to add rows to the Tag table in the database. The user just needs to input the tag name (e.g. programming) and press the “Add Tag” button to set the tag to the database.

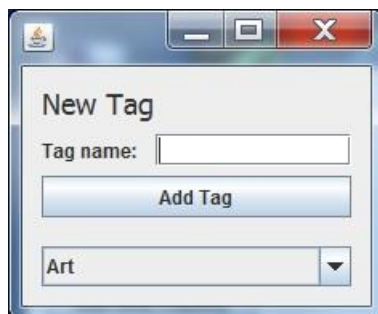


Figure 25. NewTagView

## 6 System requirements and Performance

In this chapter all of the necessary hardware and software requirements are mentioned. It is meant to inform the user of HomeLibrary what programs and hardware are necessary in order for the HomeLibrary to work.

## 6.1 User interfaces

The HomeLibrary was made on Windows 7 and should work best on Windows Vista, Windows 7 and Windows 8. Because HomeLibrary was made with Java and MySQL it should not have any problems with Unix-based or Apple operating systems, as long as the OS can support Java and MySQL. Mobile devices such as smartphones and tablets can have trouble with HomeLibrary, because the user interface was not designed with small touchscreens in mind. The performance has not been tested in other operating systems except Windows Vista and Windows 7.

## 6.2 Hardware interfaces

In addition to the computer and display, the program needs a normal computer keyboard and mouse for controlling it.

## 6.3 Software interfaces

The program connects to a MySQL database where all the data (such as book information) are stored. This database will not be visible to the user in any way other than through the program. MySQL has to be installed, as well as Java. After the MySQL has been installed, a dump file has to be run which creates the database.

## 6.4 Usage intensity

Since the HomeLibrary is meant to be installed to home computers, its usage intensity is one user at a time. Bigger intensities have not been tested.

The program can be used daily around the clock. The user is expected to make at most 10 queries per minute, on average 3. The usage intervals can be anything from 3 minutes (finding a book in the library) to 6 hours (cataloguing a library).

## 6.5 Capacity requirements

The software doesn't require heavy hardware to function. A simple modern laptop computer should be enough to handle the program and the database. The computer should have enough disk space for expanding the database.

## 7 Discussion and Conclusions

The purpose of this project was to find the best solution for managing media in a home environment. For this purpose many open source alternatives were reviewed and discarded in favor of a tailor-made system. The designing and construction of the program was then successfully implemented based on the requirements. The goal is to catalog all of Prof. Regelski's books into the HomeLibrary in the near future.

The basic features that could be included in the program during the timeframe were searching of books, adding of books and adding of other information. The editing of information was not included in this version, but is currently under construction and will be added as soon as possible. The giving roles to authors is almost completed as is the adding of bulk data from an Excel-file to the database.

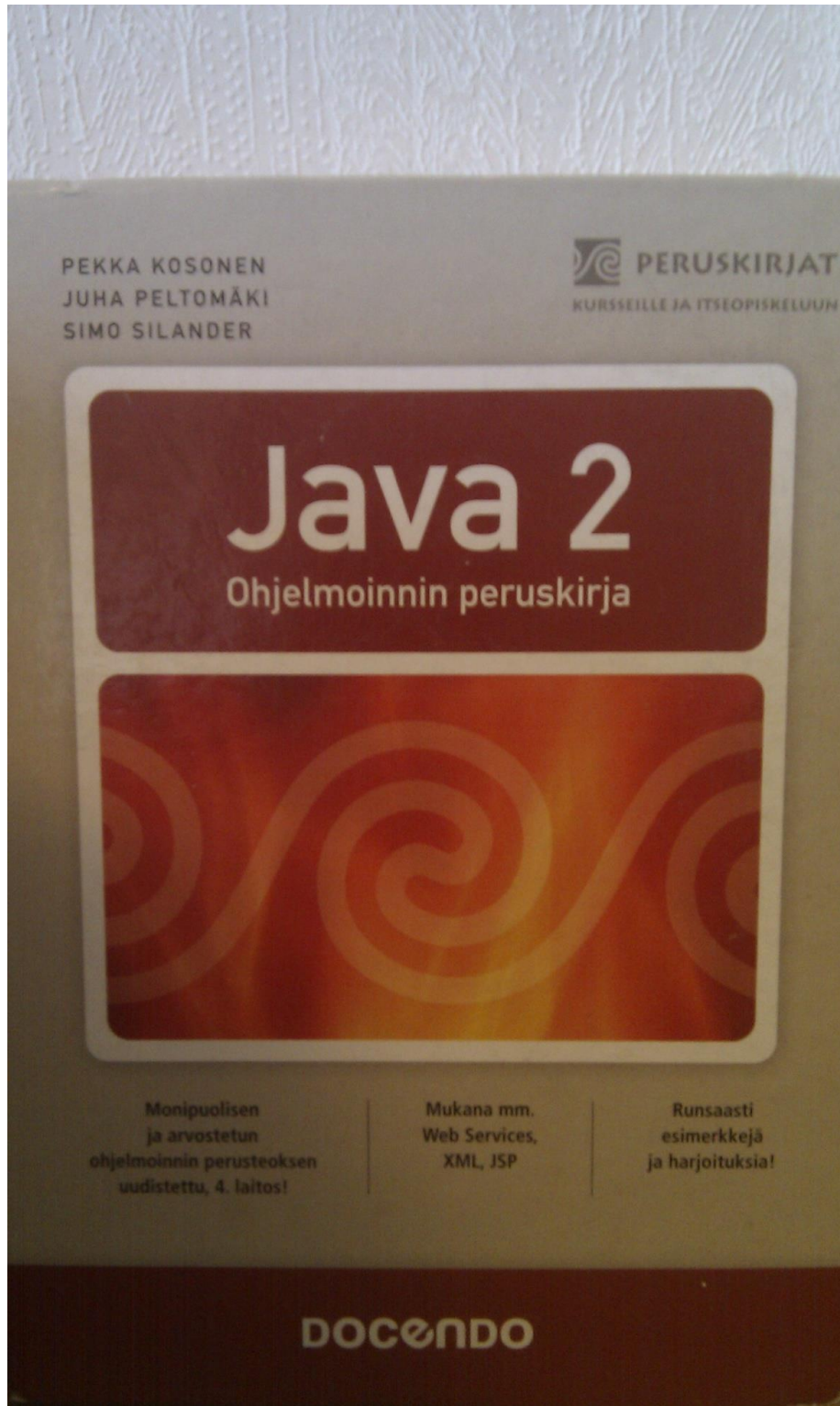
There were other features that were imagined during the project, such as automatic insertion of book information from the internet by ISBN, but time did not allow these to be implemented. There were plans for the program to make an Excel-file and export data to it, but this feature is unnecessary because the user can just copy-paste the needed information from a Java table component (jTable) to a new Excel-file. A wizard for creating the database and installing the program should be created.



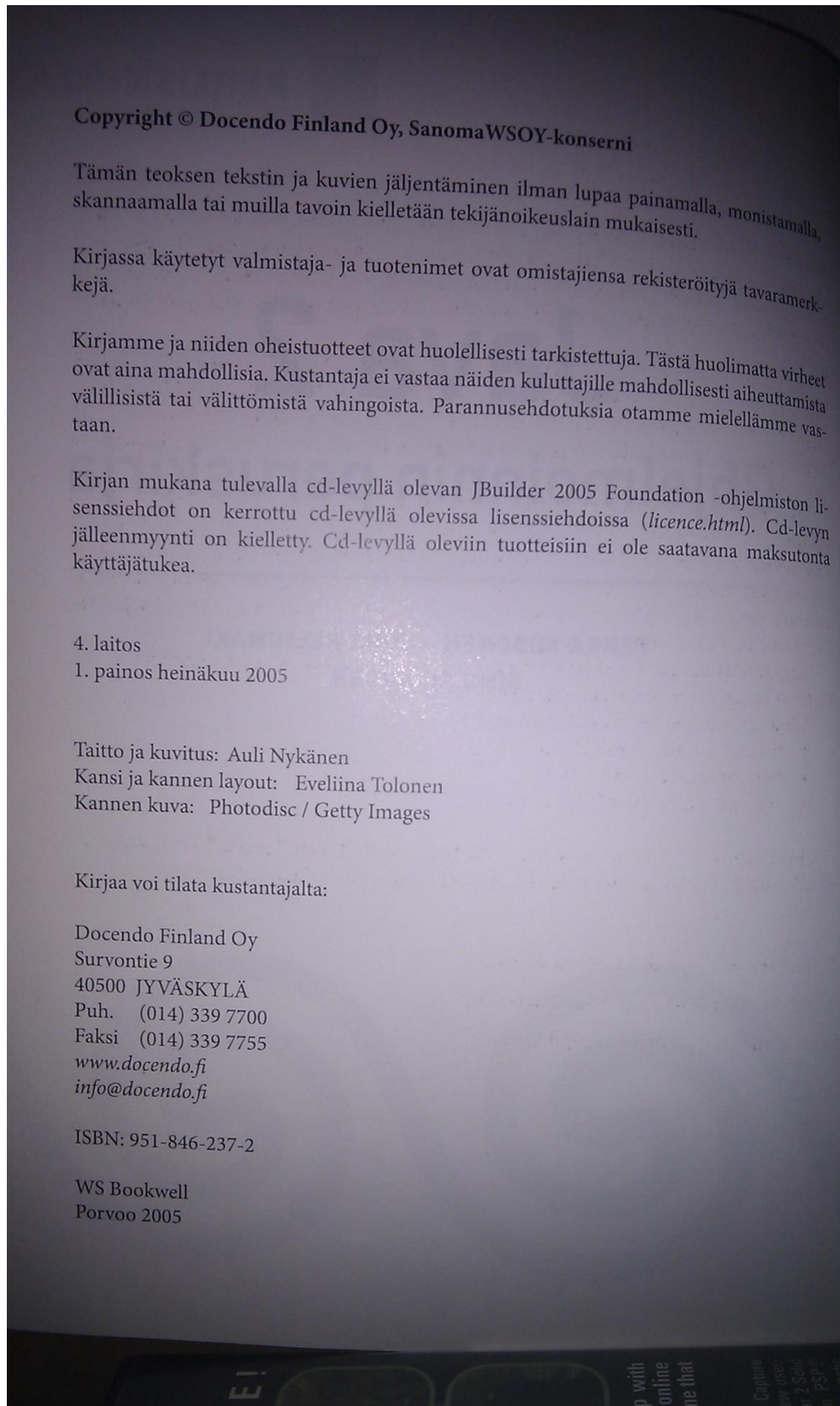
## References

- 1 Amateuring in Music and its Rivals. About the Author. Thomas A. Regelski Ph.D. Made 2007. [act.maydaygroup.org/articles/Regelski6\\_3.pdf](http://act.maydaygroup.org/articles/Regelski6_3.pdf) (20.11.2013)
- 2 About VuFind. [vufind.org/about.php](http://vufind.org/about.php) (20.11.2013)
- 3 About LibLime Koha. <http://www.koha.org/about> (20.11.2013)
- 4 About Us. <http://evergreen-ils.org/about-us/> (20.11.2013)
- 5 Greenstone (software). Wikipedia article. Last modified on 12 October 2013 [en.wikipedia.org/wiki/Greenstone-\(software\)](http://en.wikipedia.org/wiki/Greenstone-(software)) (20.11.2013)
- 6 OpenBiblio. Wikipedia article. Last modified on 31 October 2013. <http://en.wikipedia.org/wiki/OpenBiblio> (20.11.2013)
- 7 PhpMyBibli. Wikipedia article. Last modified on 16 March 2013. <http://en.wikipedia.org/wiki/PhpMyBibli> (20.11.2013)
- 8 Using Hibernate in a Java Swing Application. A Netbeans Hibernate tutorial. <https://netbeans.org/kb/docs/java/hibernate-java-se.html> (20.11.2013)
- 9 "SPECIFICATIONS DUIF DUINPAN" 05.03.2010 by Andualem Gurmu, Ilja Salmio, Elina Harjuhahto, Lauri Hynninen, Anni Lementtinen, Juha Hakala, Panu Leppaniemi, et al.

## The Example Book



Picture 1. Example Book Cover



Picture 2. Example Book Information Page