

Catalogue System for Electronic Documents Management and Control

Milena N. Karova¹, Plamena H. Grigorova²

Abstract – This paper introduces a desktop based catalogue system for managing and control of electronic documents. Electronic documents are wide spread and there is need for a tool with witch to sort, search and store these files in easily accessible and convenient way. There are few other catalogue systems for managing this kind of data but they are somewhat limited. Currently described solution extends the data stored about the electronic document while preserving the initial file, using MS SQL powered database.

Keywords – catalogue system, electronic documents, management, control, MS SQL, database.

I. INTRODUCTION

Electronic documents are currently very widely used. This is determines by their undeniable advantages in terms of necessary physical storage space, processing and searching speed and backup possibility. Electronic documents are equivalents of conventional data storage methods – books, magazines and other types of paper. A collection of electronic documents may consist of a lot more files than a real one and that's why the question of effectively naming, sorting and searching of those files on the user's hard disk arises.

II. THE CATALOGUE SYSTEM DESIGN

The Catalogue System for Electronic Documents Management and Control is simple, yet functional solution for users with collections of electronic documents, developed using C# programming language. It uses MS SQL Server to store helpful data. Its goal is to ease tasks like naming, saving additional data for the document, which is not convenient to be written in file's name, searching and opening. The aforementioned additional data may be (besides title and author) genre, encoding, comments and so called linked files, for example separate file for cover, style sheets, etc. As to authors the system allows adding their name, biographical information and picture [Fig.1], witch is stored directly into the database, thus avoiding the need to save such data separately in files on the hard disk.



Fig. 1. Author's information

The user interface is simple and familiar [Fig.1]. It's divided in three:

- Tree View [3] representing folder browsing part on the left;
- List View [3] representing file browsing part in details view on the right;
- Menus (main and context).

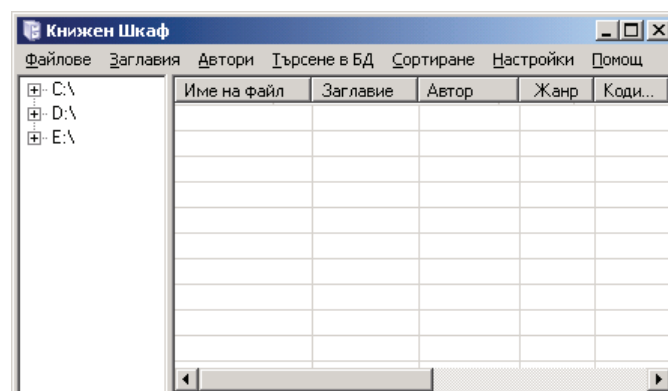


Fig. 2. User Interface

¹Milena N. Karova is with the Department of Computer Science and Technologies, Technical University Varna, Studentska str. 1, 9010 Varna, Bulgaria, E-mail: mkarova@ieee.bg

²Plamena H. Grigorova is a student within the Department of Computer Science and Technologies, Technical University Varna, Studentska str. 1, 9010 Varna, Bulgaria, E-mail: plamenagrigorova@dir.bg

The folder browsing part shows only users' hard disk drives. This limitation is made to remove obsolete records in the database. The database stores information about the electronic documents based on their location. So to avoid obsolete records, on its startup, the application runs a silent thread [2], which deletes the records without corresponding

file on the users' file system. There is no way to check database actuality on removable drives.

```
Thread t = new Thread(new ThreadStart(
    Threader.SearchObsolete));
```

The file browsing part combines typical file information like filename, type and size with data read from the database:

- Title;
- Author;
- Genre;
- Encoding;
- Number of linked files;
- Users comment.

This information is the key feature of the catalogue system, because it provides the users with the necessary data for easily managing their electronic collection.

Other very useful feature of the application is the possibility of customization included. All strings are read [1] from lang.xml file, thus allowing full translation of the system to any language. The pictures used are also read from external files and can be replaced. If those files are missing the system will simply warn the user and still run, using the only one included icon.

III. STORED DATA

The database is quite simple which contributes to the fast work of the Catalogue System. The following are the main data structures:

A. Authors' information

One author may write a lot of books and so is book independent. It's unnecessary to write the author many times for each book he wrote [5]. That's why information about them must be divided from the rest data. This information also slightly differs from the main purpose of the application, so it's very basic and suffices to following:

- Name;
- Biography;
- Picture.

The name is only one because the user may not know all the names of an author, besides some authors use one name pseudonym and this would make sorting by author difficult. The other two fields remove the need for additional files with information on authors if the user wants to have this kind of data on one place.

B. Books' information

Book information includes standard data like:

- Full filename;
- Title;
- Genre;

- Encoding;
- Comments;
- Linked files.

The full filename (like c:\folder\file.ext) is the primary key, although for faster operation in the database is used integer ID field for primary key. Title is a field that can have duplicates in the database. The reason for this is that there are books with same names and different authors and moreover – a user can have multiple books from one author which differ only by encoding. Genre is not an enumerator and is left for the users to fill in freely. The idea behind this is to give bigger control over sorting and individual view on a book's content. Encoding is int. Usually a person uses two to three languages and encoding gives a drop down choice between them, other or not readable. Those languages can too be customized in the lang.xml file. Comments are made for the user to decide what to note down. Finally linked files are those files, which come with the electronic document but are not contained in it. They include from cascading style sheets and illustrations for stored web pages to covers for books. This information is included for convenience. For example if the user decides to delete a book the linked documents information will help him clean fully his hard disk or else there may be left unused files.

IV. FUNCTIONAL DESCRIPTION

The basic tasks, performed by the application are illustrated by the following block-scheme [Fig.3]:

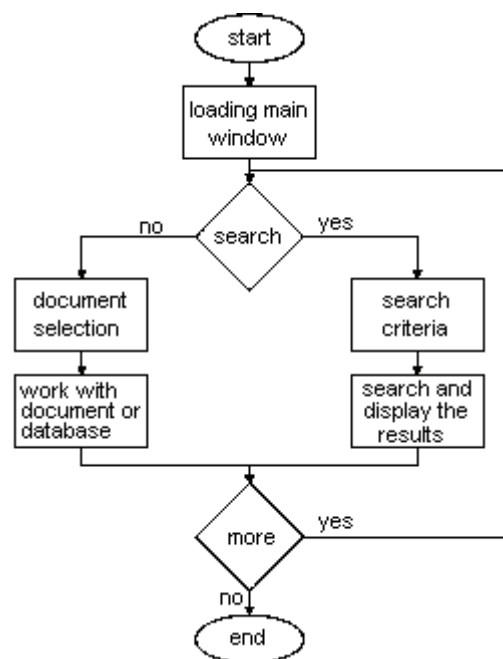


Fig. 3. Basic Functions

These problems are:

- Input of additional data for the document;
- Searching a document, based on the input data;

The first thing a user needs to do is input the data, necessary for his further work with the application. Electronic

documents can be very different in format and contents, which makes the automatic extracting of the aforementioned information nearly impossible. After input of all the needed data the user can then easily search by random criteria.

The operations that can be performed from within the system strongly depend if there is a selected file or not. If there isn't any selection some functions like opening the electronic document for reading won't be available. Actually the only available operation with data besides searching, when no file is selected, is input of author information [Fig.4].

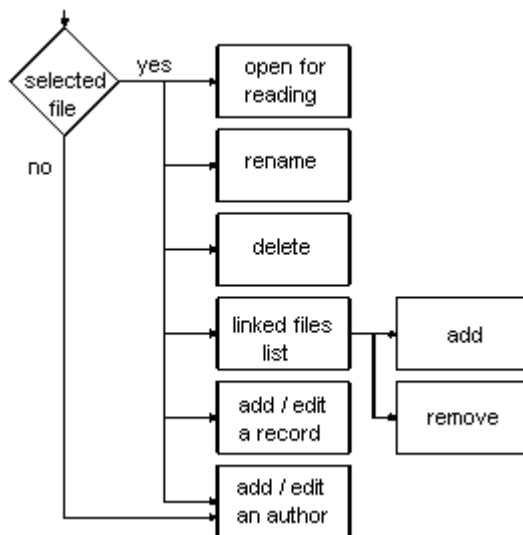


Fig. 4. File and Data Operations

The Catalogue System for Electronic Documents Management and Control has the following functions:

A. Opening a file for reading

This function calls the application, associated with the selected file:

```
System.Diagnostics.Process.Start("c:\folder\file.ext");
```

If there isn't an application associated with "ext", the program notifies the user using simple message box.

B. Renaming a file

When renaming the chosen file the system handles correctly forbidden symbols like /, \, *, ?, :, ", |, <, >, which can't be used in file names. Also if there is already a file having the new name the user is notified and the rename operation is canceled.

C. Deleting a file

When deleting a file the program checks if there is a corresponding record in the database and deletes it as well.

D. Managing linked files

Linked files are handled using a List Box. In it the full list of linked files is displayed and two buttons allow adding to and deleting from the list.

E. Creating or editing a database record

This is one of the main functions of the application, that's why "Create record" dialog can be invoked directly by double clicking on a chosen filename.

```
DialogBoxNewRecord dlgNew = new
DialogBoxNewRecord();
```

It's vital for it to be simple and compact, because the user will work frequently with it and long open times or confusing interface would be a big flaw in the system.

F. Creating or editing an author

Creating and editing an author is not connected with files, so this dialog can be invoked at any time.

```
DialogBoxNewAuthor dlgNew = new
DialogBoxNewAuthor(strPath, strFile);
```

Selecting an author for the chosen book can also be accomplished through this dialog. If there is selected file the checkbox becomes active and using it the user can select an author for his book.

One of the most important functions of the system is the possibility of searching files based on their additional data. It's independent from the selection and is invoked from the main menu. The user can search using criteria like filename, extension, genre, encoding and all the other data, stored in the database, also allowing multiple criteria selection. The multiple criteria selection however is somewhat limited as file properties (filename, extension, size) and database data are separated.

The sorting possibility can be accessed either through the menu, or simply by clicking on the ListView column' name. The last sorting order and column is saved in a configuration file, so they can be reused the next time the user starts the application.

The application algorithm is illustrated on block-scheme on [Fig.5]

After the start of the system the main constructor is executed and also the thread, which removes obsolete records from the database. The constructor creates application window, menus, TreeView and ListView controls, loads all strings from the language file if there is one and checks if the database connection is active and all resource files are present. If there is error the user is notified and default resources are loaded. Then the TreeView control is filled with users' hard disks and he can start working with the system.

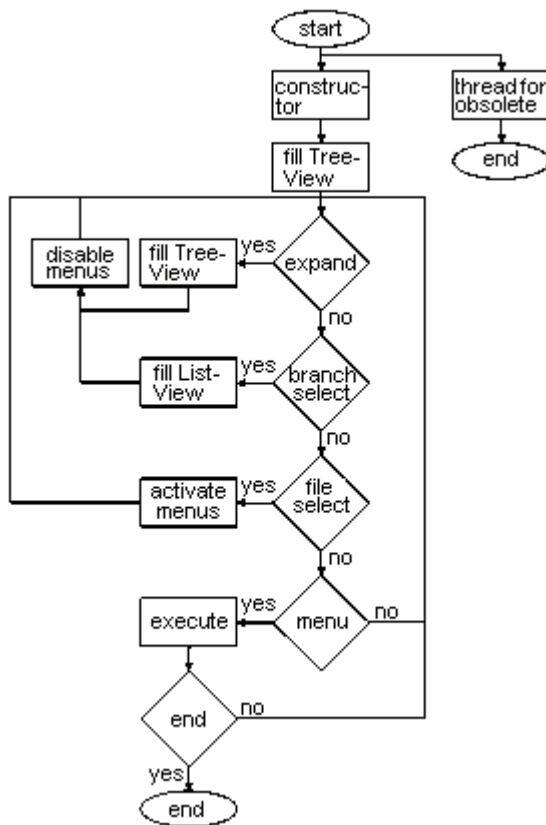


Fig. 5. Application Algorithm

Expanding a node from the TreeView calls a function which refills the TreeView. Also if there are selected files the selection is cleared and some of the functions are disabled. The same goes if a branch is selected, but then a function filling the ListView control is also executed.

If there is a selection, the functions which operate on a file are enabled. When some function is chosen it executes and if it's not "Exit" the application continues working by the explained algorithm until exited.

V. CONCLUSION

Electronic documents are useful and convenient compared to physical documents. But when their size exceeds several hundred megabytes it becomes clear that a good organization and management is needed. There are several different programming solutions for this job, but most of them store the whole document in a database limiting the user to managing their files only through that solution. If the database becomes corrupt all the documents are lost. Deleting, adding and opening of the documents goes through slow database operations. The Catalogue System for Electronic Documents Management and Control gives solution to those issues only providing management when the user chooses so, reducing database operations to minimal.

REFERENCES

- [1] Beres J., "Teach Yourself Visual Studio® .NET 2003 in 21 Days", Sams Publishing, 2003
- [2] Gittleman A., "Computing with C# and the .NET Framework", Jones and Barlett Publishers, 2003
- [3] Petzold C., "Programming Microsoft Windows with C#", SoftPress, 2003
- [4] Transact-SQL® User's Guide, Sybase 2002
- [5] <http://club.shelek.com/viewart.php?id=177>
- [6] <http://cbbrowne.com/info/edms.html>
- [7] <http://www.capterra.com/document-management-software>