DOI: 10.5937/JAES9 - 1194

Paper number: 9(2011)4, 212, 471 - 479

FROM IDEA TO IMPLEMENTATION IN PROTECTION OF ELECTRONIC EDITIONS (BOOK)

Dr Časlav Mitrović *
University of Belgrade, Faculty of Mechanical Engineering, Belgrade, Serbia
Dr Slobodan Radojević
University of Belgrade, Faculty of Mechanical Engineering, Belgrade, Serbia
Milan Srećković
Academy of criminalistic and police studies, Belgrade, Serbia
Mr Zoran Milanović
Academy of criminalistic and police studies, Belgrade, Serbia

This paper presents a study of electronic editions protection on optical discs (CD/DVD) based on basic principles of logic and côde protection. The design of this solution is rather simple, but it is mainly similar to complex methods used nowadays in protection of great number of commercial electronic editions. The implementation and usage are adapted to users (students/professors) who are interested in electronic editions that are published on Criminal-Police Academy in Belgrade.

Key words: CD, DVD, data, protective, editing, electrons, implementation, logics.

INTRODUCTION

Ensuring protection of data, information and knowledge is one of the most important and most delicate issue encountered by all social and information systems. The need for data, information and knowledge is vast, as well as the possibility of their abuse. In order to prevent their abuse and to provide system protection, different social, legal, technical, organizational and other measures are taken.

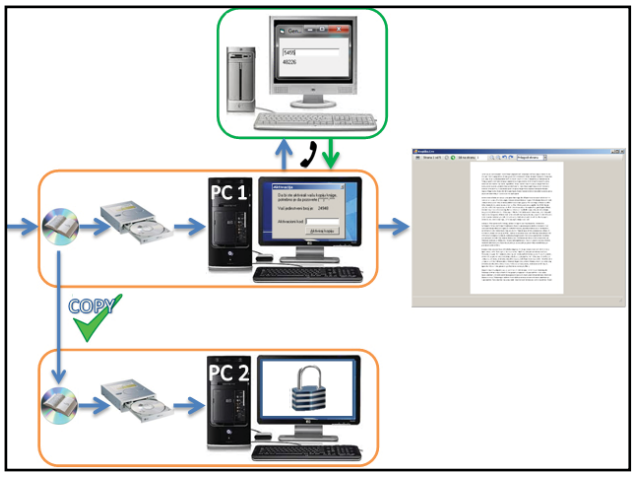

Figure 1. – Preliminary design

Nowadays the greatly accepted opinion is that the protection of information systems in information environment is something that shouldn't be foreign but domestic product. Although there is a great number of software manufactures that offer commecial solutions (Multimedia Protector [01], CD FrontEnd [02], DeployLX [03] , SerialShield [04] , CopyMinder [05]), as well as a number of websites that offer free ideas, solutions and parts of the open côde (The Côde Project [06]), we chose an example of the study in which a preliminary design of the process of logic and côde protection is given. Namely, the process can be described in few steps. The first step is provision of an electronic edition (hereinafter book) on a media, usually CD-ROM. Then there is an attempt of opening the book on user's computer. What follows is the introducing screen on which the user has choice of starting the book or the installation of the required the appropriate software for starting the book. If the user chooses to open the book, the next step is the dialogue for activation where the unique number of this computer is displayed to user, contact phone or e-mail address of the publisher or responsible person for contacting in order to

* *Faculty of Mechanical Engineering, Kraljice Marije 16, 11000 Belgrade, Serbia; cmitrovic@mas.bg.ac.rs*

get the activation côde as well as the field where the user enters the given côde. If the côde is correct, further access to the book is allowed to user. Technically, the solution is performed in the following procedure (Figure 1). The project is set in VisualStudio 2008 [7] (the language used is VisualBasic.NET) which manages of the book review. The book itself is in PDF format and put in Resource file of the project. By compiling, the book is integrated in EXE file together with the executable côde from where it is later called. During program execution the book is temporarily unpacked into temp directory from where it is taken over by the PDF file reading control which is integrated into the program. After closing the program the book is erased from the temporary location.

PROGRAM FUNCTIONS DESCRIPTION

When starting the program from the disc, the first thing seen on the screen is a menu with three options (Figure 2). It offers the book

Figure 2. – Appearance of the first window when starting media with book

opening, installation of the accompanying package and the exit from the program. The accompanying package is actually .NET Framework 2.0 [08] a set of libraries for running the programs written on VisualStudio.NET platform. This component is necessary for starting the book and the information about it can be obtained by placing the pointer over the .NET 2 button. By clicking the exit button is the path to leaving the application.In order to start the book, it is necessary to choose the first option - “Otvori knjigu”. Once the book is started, the next what appears on the screen is the dialogue for entering the activation côde (Figure 3). The explanation is here provided to the user about which phone to contact so that the activation côde could be assigned to him based on his unique number, thus enabling the starting of the book. After entering the correct activation côde, clicking the button «Aktiviraj kopiju» will make the program

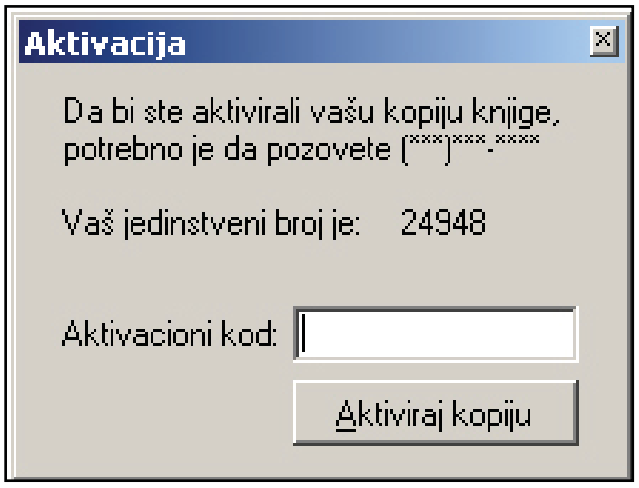

Figure 3. – Dialogue for entering the activation côde

record that number in Registry base and open the window for reading the book itself (Figure 4).

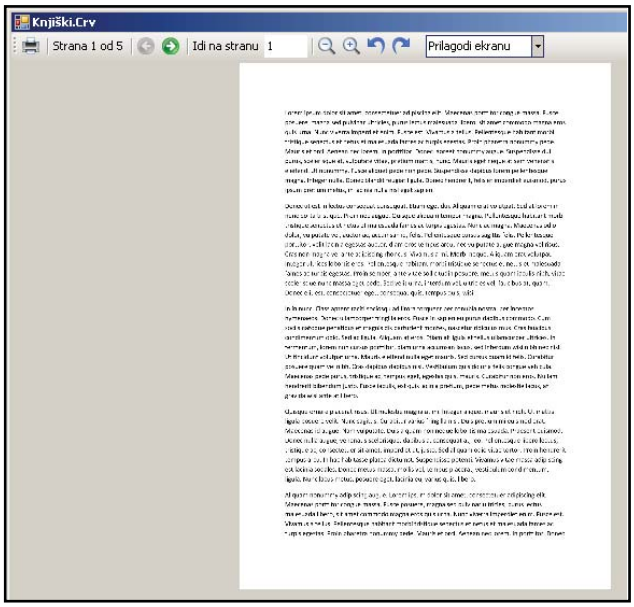

Figure 4. – View of the book

BOOK PROTECTION

Electronic book is protected on several levels. As mentioned above, PDF file is integrated into the executable EXE file. In this way, the book is not only protected from simple copying of the file, but its isolation in an independent file is made more complex. Dotsfucator [09] supplement is used during the compiling whose task is to further encrypt the EXE file and thus protects it from reverse engineering. This is another step in protecting the PDF within the program, but, what is more important, it is a very strong protection from at-

tempts of hacking into the program in order to avoid the activation procedure and try to find out the algorithm for computing the activation côde. After a successful activation, the côde is recorded in Registry database, in the section of the active user. It is thus enabled that only that user canopen the book. In multiuser systems, if the user is not in the administrator group, his right of access to write into the database is limited to only his section and that is where it is only possible to record the data. If it is necessary to write something in the section that applies to all users, the user would then be required to call the system administrator in order to start the book [10]. Furthermore, from that moment on, all the users of that computer would have the access to the book, which may not be the intention of the person who purchased it. Once the côde is recorded, it is no more necessary to enter it through the mask, yet each time when starting the program it is being checked in order to avoid the attempt of using someone else's côde by simple reconstructing the tree of Registry database from another computer. The côde itself is calculated according to the serial number of the processor on which the program is installed. Each produced processor has it own unique serial number that is added to it. That basically means that the book will be run on a certain computer until its processor is changed. A corresponding activation côde generator is on the publisher or distributor of the edition side which helps in making the côde necessary for the program activation on user's computer.

PROGRAM OPERATION PRINCIPLE-SOURCE CÔDE

The project itself consists of two sub-projects:

1. Book.Worm – program that protects the book, activates it, checks the activation côde and its recording in Registry database
2. PDFView – a set of routines, functions and libraries that perform the conversion from PDF file to TIFF image and display it in the graphics

PDFView is the part that is mostly taken from the Internet database of open côde and embedded in the unit which is used for conversion of pages from PDF files to images in TIFF format which a re drawn in a given frame (Figure 5). In this way the control is made which is later used in the main program to display the PDF file.

PDFView contains the following modules:

- GhostScriptLib – a set of functions that manage the library gsdll32.dll which contains the functions necessary for the access to PDF database and interpretation of PDF descriptive language
- ImageUtil – a set of functions for the conversion of image; manages the libraries FreeImage.dll and FreeImageNET.dll
- iTextSharpUtil – a set of functions used for accurate text rendering in the picture; it manages the library itextsharp.dll
- PrinterUtil – a set of functions for page printing
- PDFViewer – the central form that contains the control loos and which combines all the functions of this sub-project.[11]

When compiling, the sub-project PDFView is compiled in the independent library PDFView.dll which contains ready-made control that can be used in other projects and not only in this one.

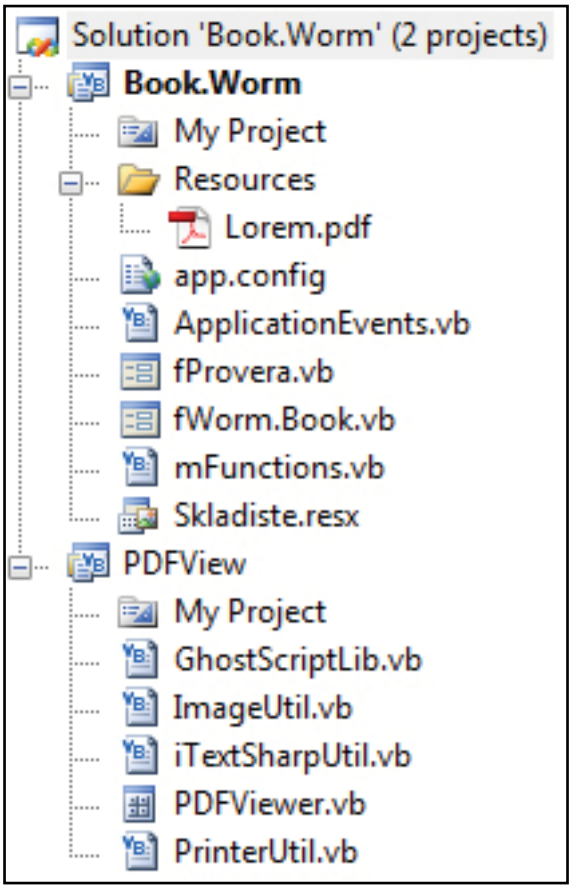

Figure 5. – Project tree structure

Book.Worm is the main project and it manages the protection logic. It contains two forms:

- fProvera – the form that verifies the activa-

tion côde if it exists, and if it does not or if it is incorrect, it requests its entering, and

- fWorm.Book – the form that contains the control which displays the book.

Beside these two forms, Worm.Book also contains the module mFunctions held by functions used in the program and Skladiste.resx resurs file where the book itself is located. Module mFunctions starts with the following two lines:

```
Imports System.Management
Public ISBN As String = “123456789”
```

A set of functions is introduced in the program in the first line which will enable to read the serial number of the processor. They are the integral part of .NET 2 environment. The second line initializes the variable ISBN string type in a value. This value is important because it is the identifier of the book. Each book has its unique ISBN number. It is useful for the data recording in Registry, because it is the way to know that the activation côde is entered for a certain book and it is impossible that two books have the same activation côde.

The module function follow.

```
Public Function fRacunajAktivacioni() As Integer
Dim A, B, C, D As Integer
A = fSaberiClanove() * 1.2
B = A * 34 + 56
C = B / 0.78
D = C - 9
Return D
End Function
```

The function fRacunajAktivacioni takes the value that fSaberiClanove function returned and transforms it through a series of arithmetic operations with the values for that book which are randomly selected in advance.

```
Public Function fGetProcessorID() As String
Dim objMOS As ManagementObjectSearcher
Dim objMOC As Management.ManagementObjectCollection
Dim objMO As New Management.ManagementObject
Dim ProcID As String = “”
objMOS = New ManagementObjectSearcher(“Select * From
Win32_Processor”)
objMOC = objMOS.Get
For Each objMO In objMOC
ProcID = objMO(“ProcessorID”)
Next
objMOS.Dispose ()
objMOS = Nothing
objMO.Dispose()
objMO = Nothing
Return ProcID
End Function
```

The function fGetProcessorID reads the serial number of the processor from the sister by means of Windows Management Instrumentation set of instructions that are embedded in Windows operating systems and forwards that number.

```
Public Function fSaberiClanove() As Integer
Dim Brojac As Integer
Dim Rez1 As Integer
Dim Rez2 As Integer = 0
Dim ProcID As String

ProcID = fGetProcessorID()
For Brojac = 1 To Len(ProcID)
Rez1 = AscW(Mid(ProcID, Brojac, 1))
Rez2 = Rez2 + Rez1
Next

Return Rez2 * 28
End Function
```

The function fSaberiClanove converts each individual symbol from the processor serial number in the ASCII value, then adds them and in the end multiplies the sum with the value for that book which is randomly selected in advance.

```
Public Function fSnimiAktivacioni(ByVal Broj As String)
SaveSetting(“Knjiga”, ISBN, “Aktivacioni”, Broj)
Return 0
End Function
```

The function fSnimiAktivacioni puts the correct activation côde in Registry.

```
Public Function fProcitajAktivacioni() As Integer
Dim SSer As String
Dim ISer As Integer
SSer = GetSetting(“Knjiga”, ISBN, “Aktivacioni”)
ISer = Int(Val(SSer))
Return ISer
End Function
```

The function fProcitajAktivacioni reads the activation côde from Registry database and forwards it. These are the main functions that are performed either by calling each other or by being called from other places during the execution of the program.

The first place encountered when the program starts is the event fProveraLoad(). It contains the following instructions.

```
Me.Visible = False
If fRacunajAktivacioni() = fProcitajAktivacioni() Then
fWormBook.Show()
Else
Me.Visible = True
lJedinstveniBroj.Text = fSaberiClanove()
End If
```

The form is hidden from the user in the first line. Then the comparison of the calculated activation côde and the one read from the database is performed. If the database does not contain any data, the function fProcitajAktivacioni will return the value 0. If it is determined by comparing that those two numbers are the same, the program will open the next form for viewing the book. Otherwise, a dialogue with the information necessary for activation will be displayed. After contacting the technical support and obtaining the activation côde, the user is expected to press the button “Aktiviraj kopiju”. It is covered by the event bAktiviraj_Click().

```
If tAktivacioniKod.Text = fRacunajAktivacioni() Then
fSnimiAktivacioni(tAktivacioniKod.Text)
fWormBook.Show()
Else

MsgBox(“Uneti broj nije ispravan!” + Chr(13) + “Pokušajte
ponovo.”, MsgBoxStyle.Exclamation)

End If
```

This event starts with confirming the activation côde that the user entered. If the côde is correct, the program will call the function that records it in Registry, and then the book will open. Otherwise, the user will receive the message about error.

The form that opens the book contains three events: fWormBook_Load(), fWormBook_Resize() and fWormBook_FormClosing; and one function – PrilagodiVelicinu.

```
Private Sub PrilagodiVelicinu()
PdfViewer1.Location = New Point(0, 0)
PdfViewer1.Width = Me.Width - 15
PdfViewer1.Height = Me.Height - 40
End Sub
```

The function PrilagodiVelicinu serves for adopting the size of the container PDFView control to the size of the form so as to touch its edges from the inside.

```
fProvera.Close()
PutanjaFajla = IO.Path.GetTempPath & “B.pdf”
IO.File.WriteAllBytes(PutanjaFajla, My.Resources.Skladiste.
Lorem)
PdfViewer1.FileName = PutanjaFajla
PrilagodiVelicinu()
```

This is the body of the event fWorkBook_Load(). In the first line the program closes the previous form. Then the value of the variable PutanjaFajla is formed which contains the path to the file where the book will be temporarily unpacked. This location in within the implied Windows Temp directory, and the name of the file is b.pdf. In Windows operating systems that are based on NT technology, the implied temporary directory is formed for each user separately and is usually located on the path: c:\documents and settings\user\local settings\temp. What follows is the drawing the book in user’s control and adopting its size to the dimensions of the form.

CONCLUSION

When estimating the degree of protection accomplished, one should bear in mind that the absolute protection cannot be achieved, but it is necessary to strive for maximum safety at any time. In that sense any real protection system requires continuous upgrading and improvement, regardless of the quality achieved so far. Corrective actions should be focused to modifications of the solutions with noticed weaknesses to the acceptable satisfactory solutions, or their elimination from the system in case of impossible modification; as well as finding, de-

veloping and applying of individual original solutions for protecting, as it is shown in this work. The practical contribution of the paper is reflected in the design and implementation of new technical solutions to protect electronic books, which is now increasingly used in scientific research and educational institutions. This solution is not prevented from copying a physical media on which the issue is, but it is practically difficult to use it without the appropriate activation procedures.

This type of protection is more interesting and „rigid“ than the physical protection of media. Naimely, nowadays it is easy to find tools that very proficiently emulate different mechanisms for media protection, and the procedure with the activation of software is still difficult to avoid, partly because it is unique for each edition and partly because of its complex procedure. The solution shown in this paper is only the example of protection technology used. For producing it, it is necessary to introduce some more improvements. Another possible way of improving it to introduce the serial numbers. In this way the monitoring of each individual edition would be enabled as well as the number of activation performed. The second way is to introduce more complex algorithm for côde calculation. The third way is to provide the activation over the Internet, but the implementation of serial numbers is necessary for this step. In more advanced stage of development, a cross-platform solution could be an option, that is a software that could run on multiple systems (Windows, Linux, OSX,…).

REFERENCES

1) Mirage systems: All-In-One Protector (http://www.mirage-systems.info/download/mp/ marketing/ flyer-multimedia-protector-en.pdf)
2) http://www.cdfrontend.com/cd_dvd_presentation_brochure_00002e.htm
3) http://xheo.com/products/copy-protection?gclid=CJvszfqdq6YCFY0r3wodg2w8pA
4) http://www.ionworx.com/serialshield.html?gclid=CNnw06KNoaYCFYUw3wod9G-goQ
5) http://www.microcosm.co.uk/copyminder.php?gclid=CLj8j8KSoaYCFUco3wodBVQ9oA
6) http://www.côdeproject.com/KB/applications/opensrcprot_part2.aspx
7) Visual Studio 2008, http://msdn.microsoft.com/en-us/vstudio/default.aspx
8) .NET Framework 2.0, http://msdn.microsoft.com/en-us/netframework/aa731542.aspx
9) Dot Fuscator, http://www.preemptive.com/products/dotfuscator/overview
10) Mitrović, Č., Vorotović, G. Modeliranje informacionog sistema za praćenje sastava i načina eksploatacije pneumatika u vazduhoplovstvu. Journal of Applied Engineering Science, 1 (2003) 2, 35-52.),
11) PDF Viewer Control Without Acrobat Reader Installed, (http://www.côdeproject.com/KB/ applications/PDFViewerControl.aspx?msg=3127440)

Paper sent to revision: 15.11.2011.

Paper ready for publication: 01.12.2011.