LULEÅ UNIVERSITY OF TECHNOLOGY

1999:50 ING

BACHELOR'S THESIS

Set-up application for self-service machines

Jan Pasanen

BACHELOR OF SCIENCE PROGRAMME
COMPUTER ENGINEERING

Skellefteå Campus

1999: 50 ING · ISSN: 1402 - 1609 · ISRN:LTU - IP - EX -- 99/50 --SE

Preface

This Bachelor thesis includes a study of old documentation on the set-up of the self-service machine CDS / CDP 700, and a description of a set-up application implementation. The work was carried out at the Division of Computer Science, Luleå University of Technology, Skellefteå Campus.

I would like to thank my supervisor Anders Gustafsson, for his help with all material and good knowledge concerning the self-service machine.

A very special thanks goes to Bengt-Arne Fjällner, the technician of the institution, because of his excellent programming skills and deep knowledge in Windows programming.

Finally I would like to thank my lovely wife for her understanding and enormous support during my studies and this Bachelor thesis.

Skellefteå, October 1999

Jan Pasanen

Abstract

This thesis report will inform you on how parts of a set-up application are developed for a self-service machine. The final result is the application itself, a description on what parts of the implementation that can be reused and a short description on how to use the application.

Table of Contents

1 Introduction

1.1 Background

Self-service machines, especially cash withdrawal machines, are something that people get more and more used to. Deposit machines, on the other hand, are not as common as the above-mentioned machines, but have been out on the market for a long time. During the last 10 years, with all the inventions and prospering on the electronic market, these machines have developed from straight mechanical ones into a very sophisticated mixture of mechanics and electronic devices. Behind this evolution stand the demands from the finance and banking companies that has come a long way with the information technology. Because of this the recent models of self service machines are equipped with a PC, which makes it possible to integrate the machine into the existing network of the company.

With new electronic components and in fact a full-blown PC in a machine, the software that controls and manages the flow of the machines has become larger and more complex. Just the step to provide a new operating system, as in this case Windows NT, will put more demands not only on the developer, but also on the technical service support.

1.2 Aim and Purpose

The purpose of this Bachelor thesis is to develop parts of a set-up program to the self-service machine software. The software shall run under Windows NT (sp4[1]). The graphical user interface, GUI, shall be developed in co-operation with the department of technical support at SCAN COIN AB.

Lots of similar applications for other products and a newly initiated set-up program are taken into consideration during development. The Forms Model in the XFS[2] interface specification is taken into consideration during development of the receipt set-up. The number of modules that are provided with set-up possibilities are limited during work to suite a Bachelor thesis.

Requirements

- Definition and development of suitable structures to store set-up.
- Development of the set-up-GUI for a number of modules.
- Development of a help function to show the possibilities.
- Development of the format of the set-up for receipts.
- Development of possibilities to test the set-up of the receipts.
- Development of the format of the set-up for pictures and sound.
- Development of possibilities to test the set-up of the pictures and sound.

Result

The result is a demonstration application, and a description on how to add modules and functions to the set-up program. Reusable components are also documented.

1.3 Limitations

Due to that this thesis was intended for two persons it had to be reduced in some way. The picture below describes the aim in the beginning of the work.

[1] Service Pack.
[2] See 1.5.

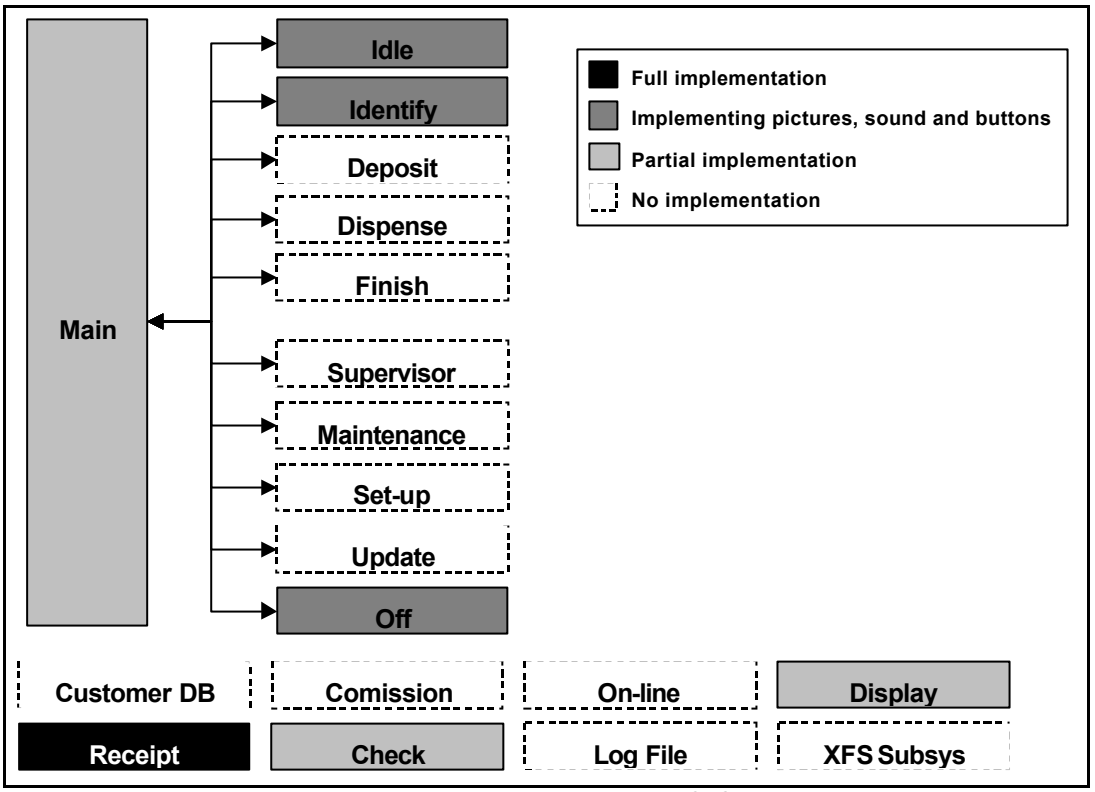

Figure 1.1 these are the modules that will be implemented in the project CDSWinNT. The set-up application will implement set-up for the modules that have a solid frame.

The following limitations were made during work:

- The design and implementation of the receipt module are left out. The specification of the set-up remains.
- The help file function is implemented partly and no proper help text is provided.

1.4 Extensions and clarifications

Project management

During thesis work I discovered that it would be suitable to store a set-up as a project. A new project is created by choosing which existing project that will be copied. This will dramatically speed up the set-up procedure. Because of this there is always at least one existing project – the default project – that holds a common set-up.

Import and export projects

To make it easy to send a new set-up to a customer an export function was added that compresses the project into a ZIP-file. This will also make it easy for the customer to send his current set-up to technical support. Because of this, an import function was added, to serve both the customer and the technical support.

Picture

I decided to use the name slide instead of picture because the slide itself can contain one or more pictures.

Set-up

Some modules have two types of set-up:

- Visual
- Non-visual

The visual set-up describes how the different slides of the modules look like. The non-visual describes some kind of setting for that module.

1.5 XFS

The XFS specification is an open architecture used as a middleware between the application and hardware drivers in banking and financial systems. One requirement was to take XFS interface specification Part 3: Printer Device Class Interface under consideration when developing the set-up for the receipt.

Origin

This specification was originally developed by the Banking Solutions Vendor Council (BSVC), and is endorsed by the CEN/ISSS Workshop on XFS. This Workshop gathers both suppliers (among others the BSVC members) as well as banks and other financial service companies.
Copied from CWA 13449-3 XFS interface specification Part 3. [3]

Printer Device Class Interface

One part of this specification contains the Forms Model[4] and Media Definition[5]. These parts have been completely examined and used when defining the INI-file format for the Receipt module.

[3] http://www.cenorm.be/isss/Workshop/XFS/cwa13449/Default.htm
[4] A general script language describing how a form is designed.
[5] A general script language describing the limits of the media to print on.

2 Methods

2.1 Collecting information

My supervisor provided me with all the necessary textual information, concerning the new software that was under development. Much of the information in the field of programming I got from Borland C++ Builder's help and by searching the Internet.

2.2 Set-up structure

The INI-file format is used to store the set-up persistently. Another possibility, binary file, was discarded early in this work, most because of the great support for the INI-file format in Borland Builder. Another benefit is that this format is plain text and therefore easy to maintain and read without having to use an advanced application, as would have been the case if a binary file was used.

2.3 Help file compilation

The software Robo Help Office was used to make the creation of the help file easier. This tool is used seamlessly with Microsoft Word, giving the user the opportunity to write the whole help file in word and then compile it into different help file formats. There are mainly 3 different formats, WinHelp, Microsoft HTML Help, and Cross-platform Help. After a while I chose to use the common WinHelp format.

2.4 Implementation

The set-up application is written in C++ using the Borland® C++ Builder 3 (Borland Builder) environment. During the whole implementation consideration was taken to make the code as reusable as possible.

3 Module description

Each module represented in the CDSWinNT application will be explained here.

3.1 Main

This module is the main application that will manage the module DLLs and distribute messages among them.

3.2 Idle

The idle module handles the application program flow during an idle cycle, i.e. waiting for a customer.

3.3 Identify

The identify module handles the application program flow during customer identification.

3.4 Off

The off module handles the application program flow during an out of use state.

3.5 Check

The check module handles for local verification of the card and account numbers entered by the customer.

3.6 Receipt

The receipt module handles receipt printing for both customer and maintainer.

3.7 Display

The display module handles all the graphic output on screen and sound.

3.8 Set-up

The set-up module handles the set-up of the CDSWinNT application.

4 INI-files

Each module has one INI-file, with a GENERAL section as follows. The set-up application uses the TMemIniFile class provided by Borland Builder to get access to the INI-files.

[GENERAL]
Program=CdsNTModuleName
Version=XX.XX

Because of this general section a generic class, CIniFile[6], was made that has Set and Get functions towards these settings. All the INI-files are listed in Appendix C.

[6] See Appendix A for more information about the class.

5 GUI

The GUI is built up as shown in Appendix B. I used a tree view to navigate among the different modules. For each module there is a child window with the set-up possibilities. In the display module there are several child windows, one for each slide.

5.1 Slide set-up

The set-up for each slide is rather complex, but made easy to the user by doing it graphically. See 0 for further information.

6 Help

Each part in the GUI that need to be explained in some way will have a help page linked to it. For now it is only the main window, the tree-view and the check window that have help implemented. (Just to show that it works)

7 Application functionality

The set-up application works as follows.

7.1 Start-up

A valid password must be provided to the application to enter the set-up. This password is encrypted to protect the set-up. If the password is correct you have the possibility to choose an old project or create a new one.

7.2 Project handling

Existing project

If you choose to change an existing project the main window is shown and the set-up can be changed.

New project

If you choose to create a new project you must provide a name for the project and then choose which project the new project should be based on. This means that the existing project set-up will be copied.

Closing a project

To close the open project start with closing all the child windows, if any open, then click Close project in the File menu. If you directly choose Close Project in the File menu, the set-up application will crash due to a bud that has not been fixed yet.

Deleting a project

To delete a project you need to get to the project dialog. Do this by choosing the Open project in the File menu. Choose the project you want to delete and click the Delete button.

Exporting a project

Choose the File menu and click Export project to create a ZIP-file that contains the project. This operation uses the PKZip version 2.50 and also supports disc spanning if you would like to have the project on disc.

7.3 Changing set-up for a module

To change the set-up for a module you just have to double click the module name in the tree-view on the left. This will load the module DLL and show the set-up possibilities for that module. The INI-file for the module is read into memory when loading the DLL. When you change a setting the change is only made in memory until the Save button is clicked.

Change set-up for all modules except display

To get further information on the set-up possibilities for these modules take a look at Appendix B.

Change set-up for display module

The set-up of a slide is rather complex. The goal was to make easy by making the set-up in a preview mode. It's possible to add graphic objects to the slide with some limitations:

- The maximum numbers of buttons are eight, due to hardware limits.
- Only one sound per slide.

Each object has some kind of set-up possibility. Just right click on an object and a popup menu will appear with all options. From there you can align and change the object but also delete and add objects. The alignment of buttons has an extra possibility to align it to the fixed buttons on the real machine, making it easy to get the buttons at the right positions.

Each object is movable and sizeable. It is also possible to have transparent bitmaps. Note that transparent in this case means that one colour is replaced with transparency. If the pictures topleft pixel is the transparent

colour the application will treat that colour as the transparent color for the rest of the picture. To tell the application that the top left pixel is the transparent colour the transparent checkbox in the button- / image settings dialog must be checked. If that pixel is not the transparent colour, then the transparent colour must be choosen from the palette.s

To get further information take a look at Appendix B.

Saving the changes

To save the changes you just click the save button. Note that this will close the child window of the module. This operation feels strange and therefore this button will not close the child window in the next revision of the application.

Discarding changes

To discard changes made just click the cancel button in the child window. Note that this closes the child window. The Cancel button text will be replaced with the text Close, to make it more clear what the button does.

7.4 Changing password

To change password just choose the Settings menu and click Change password. Then enter the old password and the new password twice, and then click OK. Note that this password is encrypted in the INI-file and do not forget to write it down somewhere!

7.5 Managing multiple child windows

You can have multiple child windows open at the same time. To manage those windows, just use the Window menu. There are 4 options in the menu: Previous, Next, Cascade and Tile.

8 Reusable code

All classes are made as reusable as possible. Here are some suggestions on reusable code.

8.1 Graphic objects

All the graphic objects developed can be used to display the slides in the CDSWinNt application. Read Appendix A to get information on the classes involved.

9 Discussion

This thesis was, as I stated before, intended for two persons. I tried to limit the thesis in the beginning, but underestimated the size of it. Maybe if I had kept to the thesis specification and requirements I would have made this on time. Some changes and extensions were made that are mentioned before. The main thing that delayed me was the user interface for the display module. I tried to make it more or less like Microsoft Power Point, with draggable and sizeable textfields, buttons and images. Another major problem I had was to make the project management work. I use PKZip and that was the first time I ever tried to start another program from another application.

The distance between Malmö and Skellefteå made it hard to get the right answers from my supervisor, although I went there regularly.

10 References

10.1 Documents

Software design specification CDS/CDP Windows NT.

Technical specification CDS/CDP Windows NT

CWA 13449-3 XFS interface specification Part 3

10.2 Internet

www.borland.com – both discussion groups and FAQ.

bcbdev.com – BCB Dev, extremely good FAQ's.

www.vb-world.net/features/shfileop – VB-world good introduction on using the new file copy dialog.

10.3 Other

Borland Builder Help.

Win 32 API SDK Help.

11 Appendix A – Class descriptions

11.1 Display specific classes

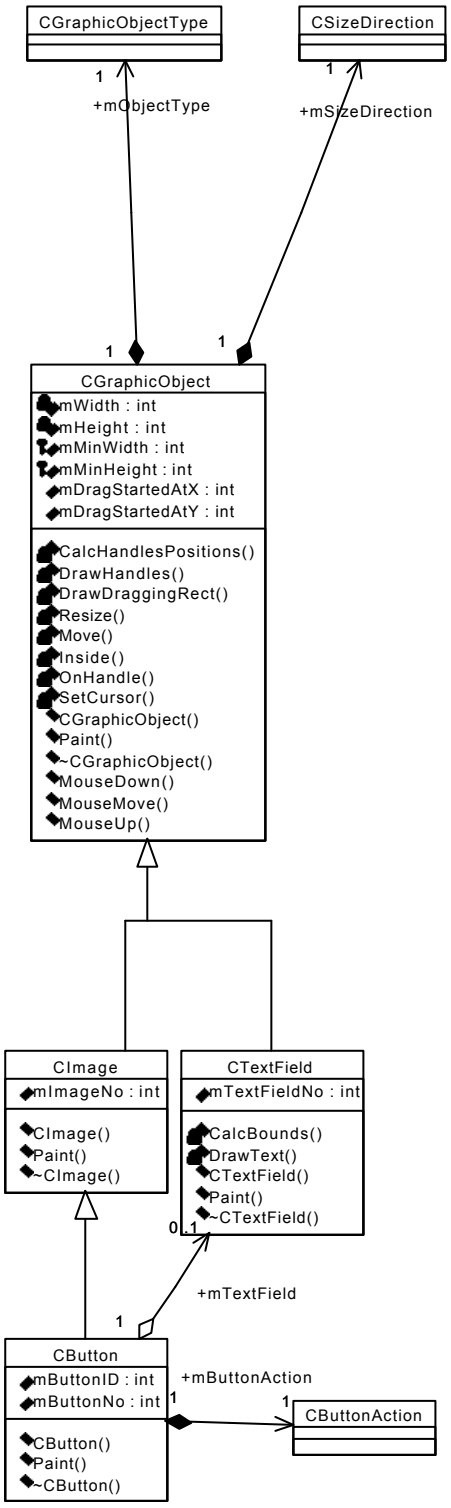

Figure 11.1 Shows the objects used to display the graphics in the display module set-up.

11.2 Other classes

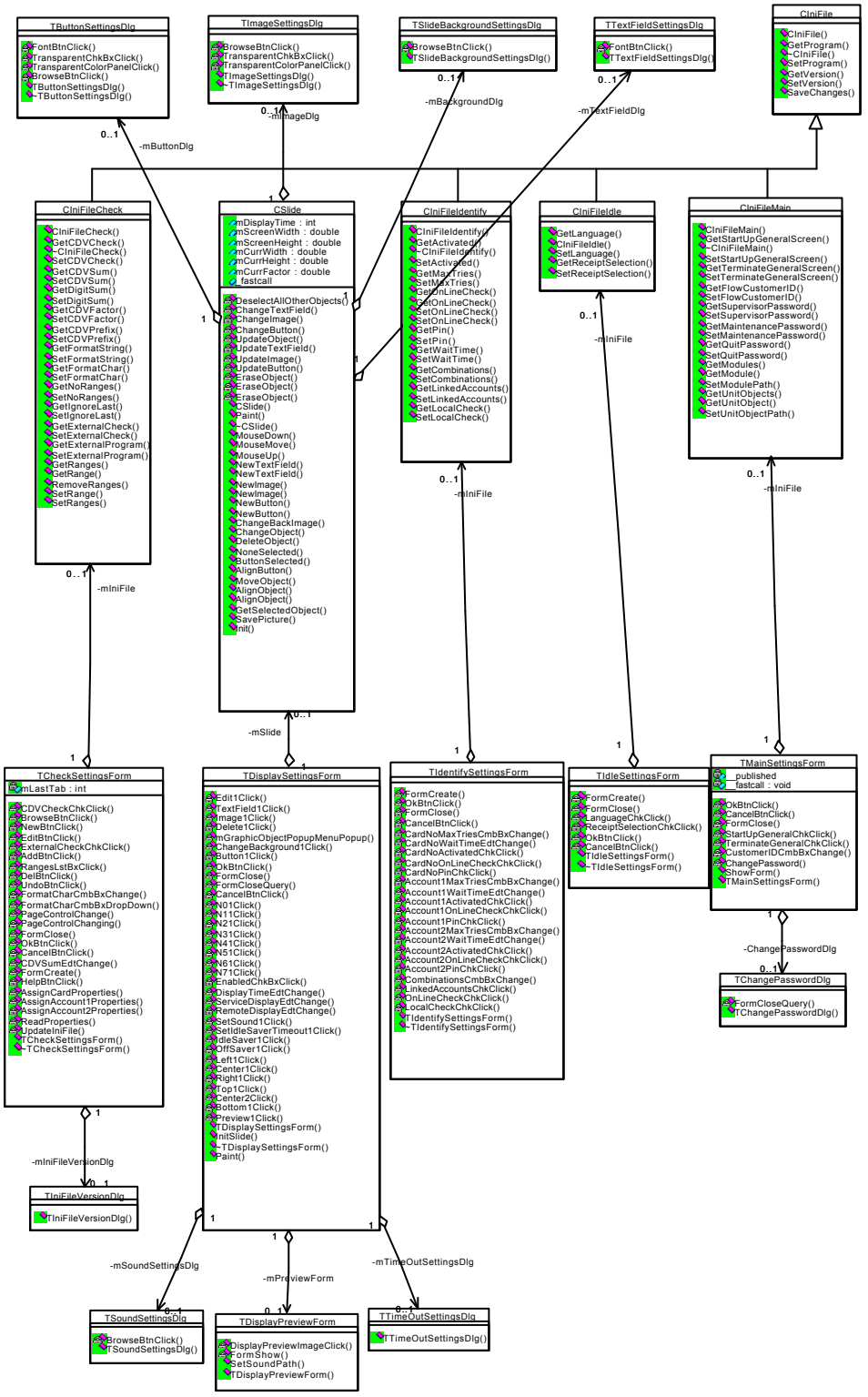

Figure 11.2 Shows all the other classes that I developed.

12 Appendix B – GUI with explanations

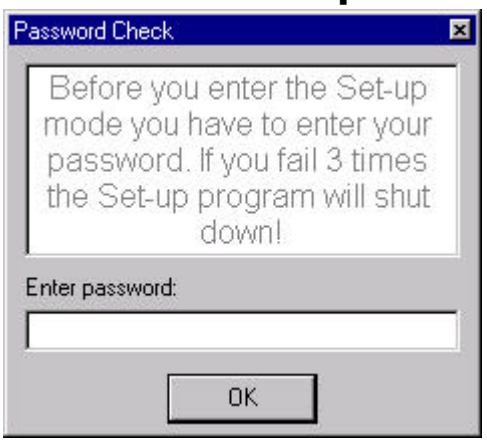

Figure 12.1 shows the Password Check dialog that appears at start-up of the set-up application.

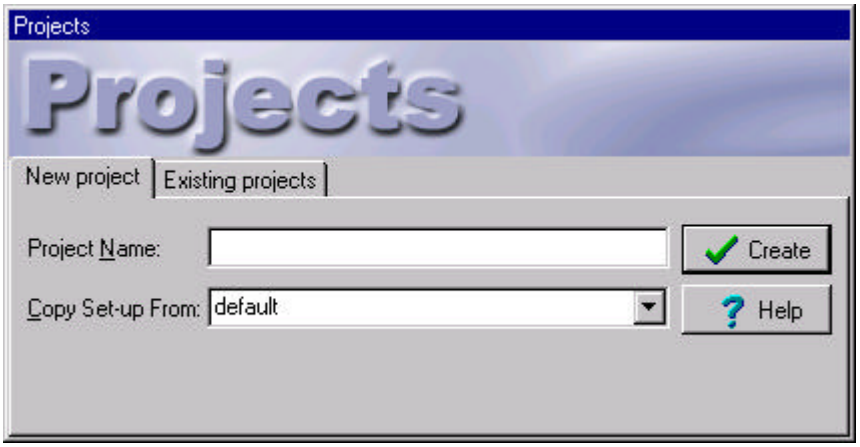

Figure 12.2 shows the project management dialog and here the new project tab.

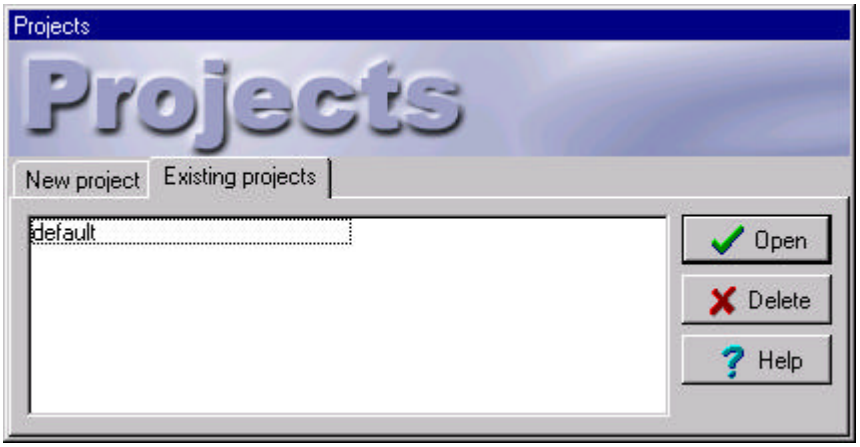

Figure 12.3 shows the project management dialog and here the existing project tab.

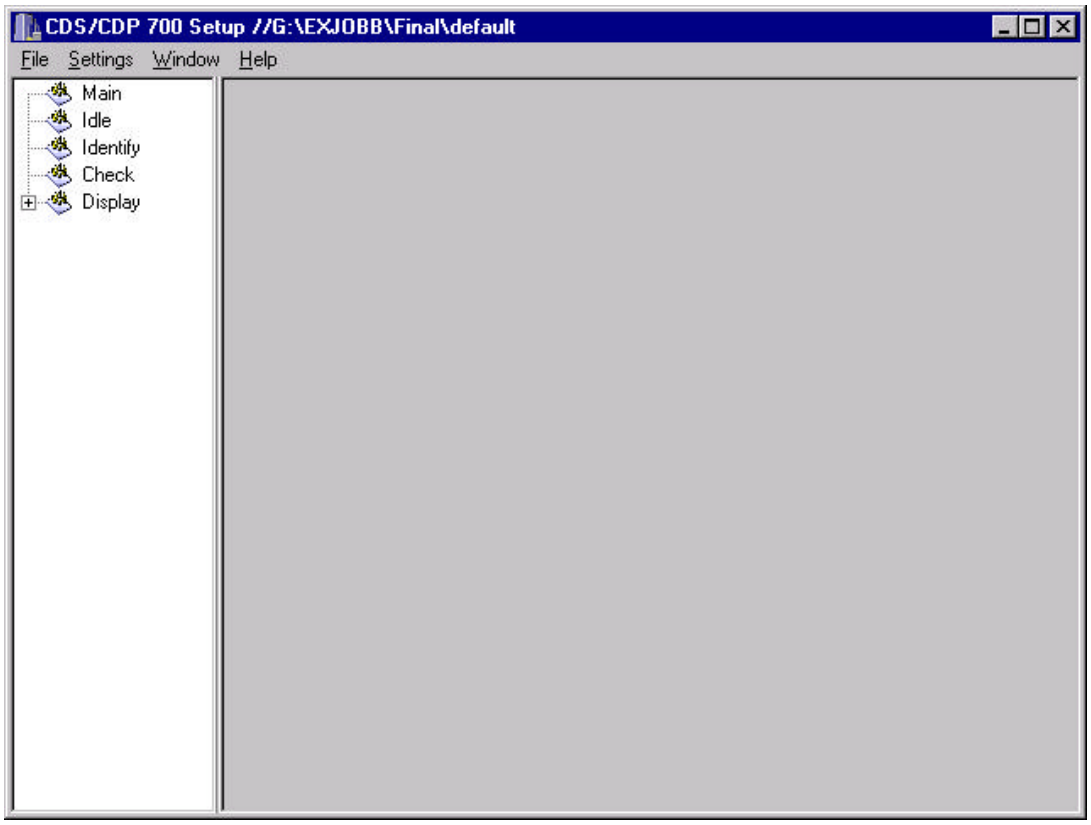

Figure 12.4 shows the main window after selecting an existing project.

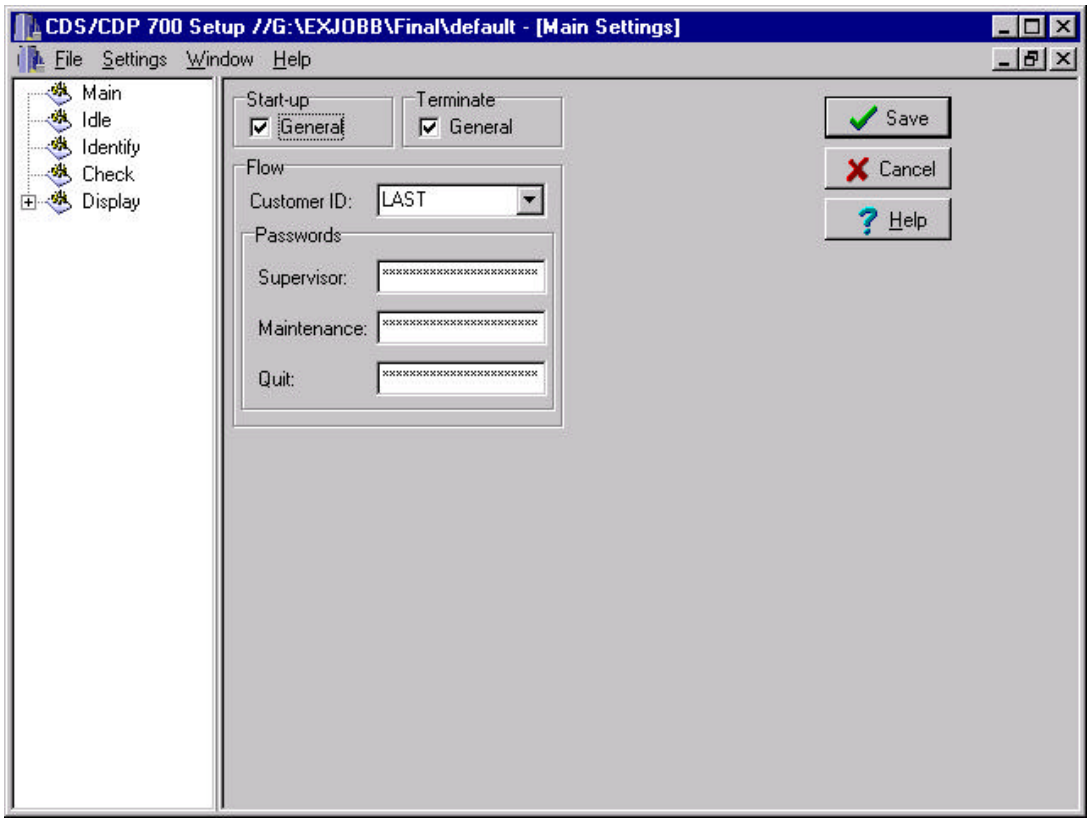

Figure 12.5 shows the set-up possibilities for the main module.

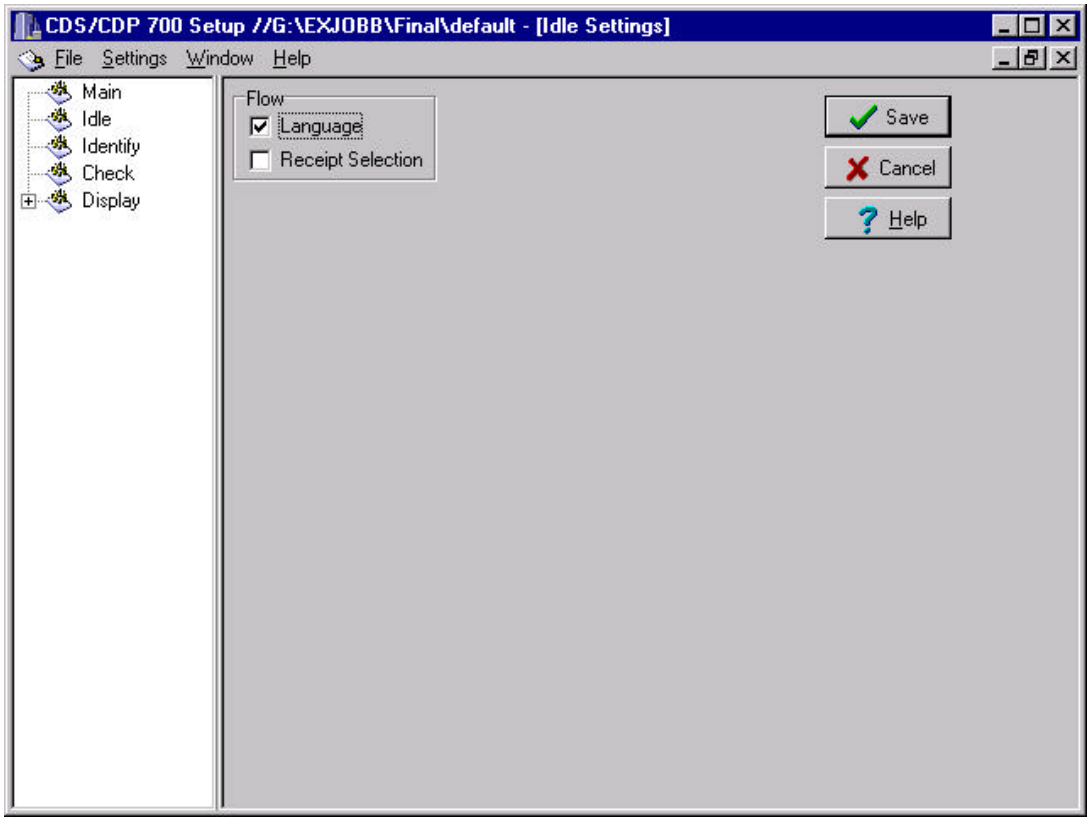

Figure 12.6 shows the set-up possibilities for the idle module.

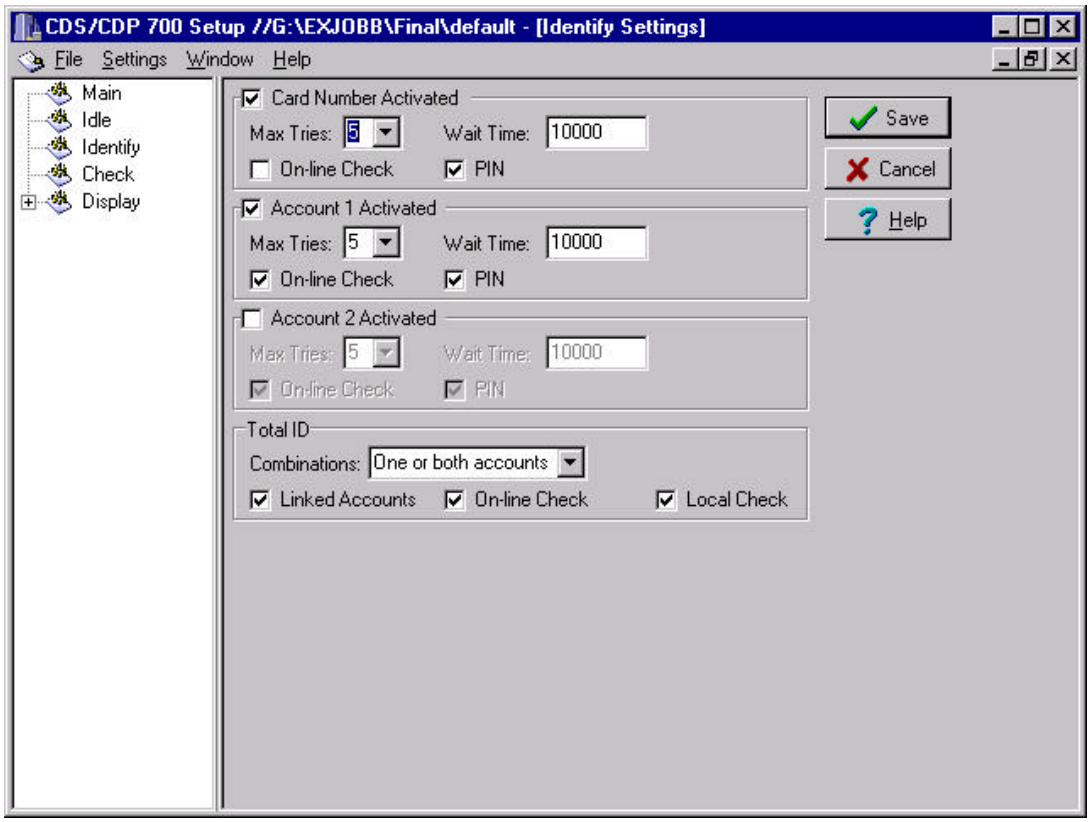

Figure 12.7 shows the set-up possibilities for the identify module

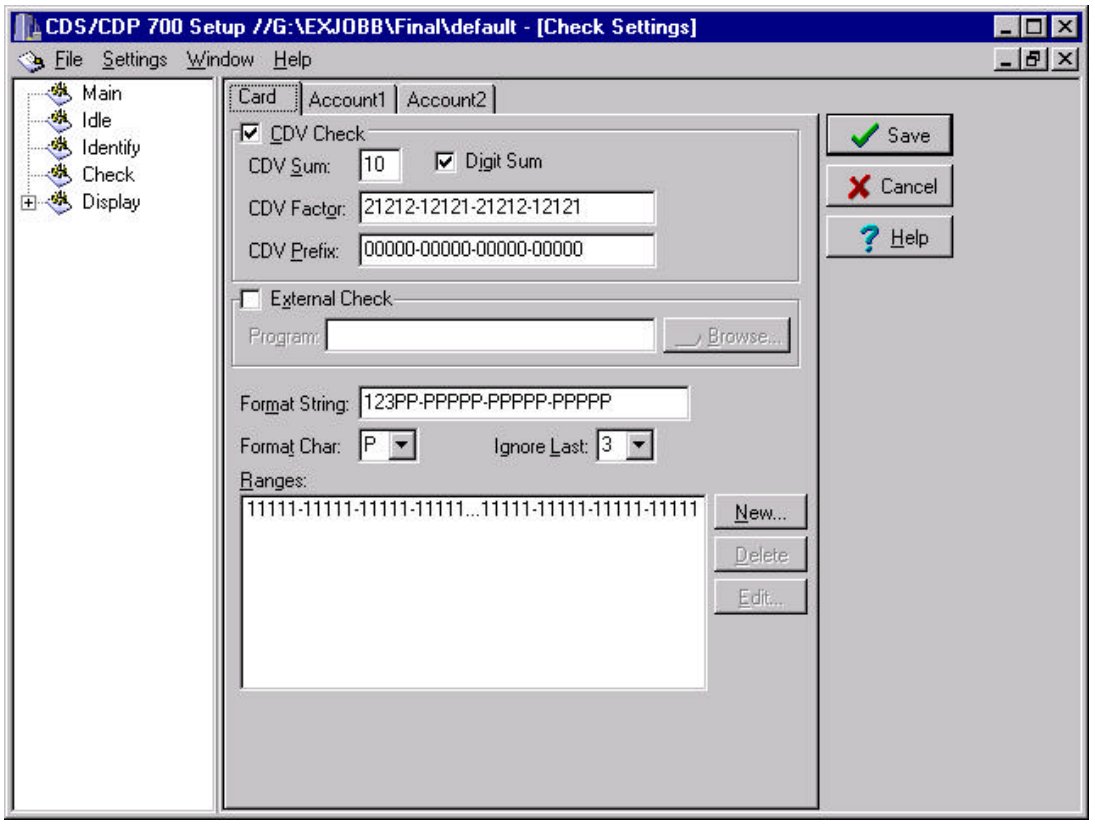

Figure 12.8 shows the set-up possibilities for the check module.

Figure 12.9 shows the set-up possibilities for one slide.

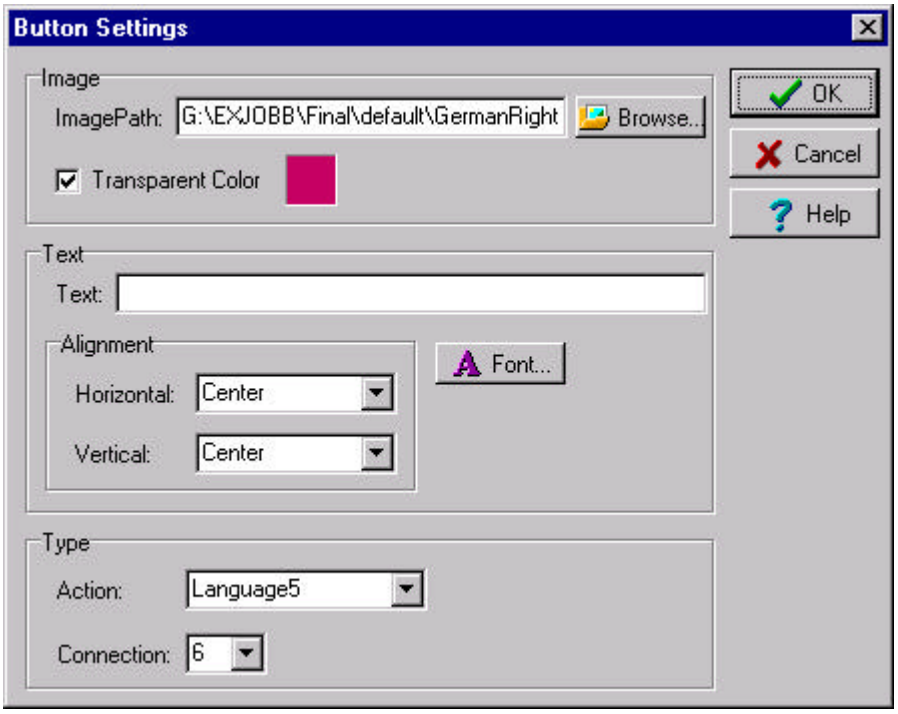

Figure 12.10 shows the button settings dialog, with all the possibilities that are present.

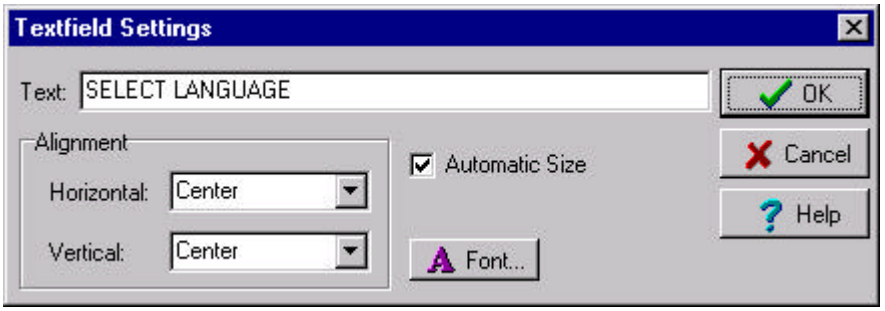

Figure 12.11 shows the textfield settings dialog, with all the possibilities that are present.

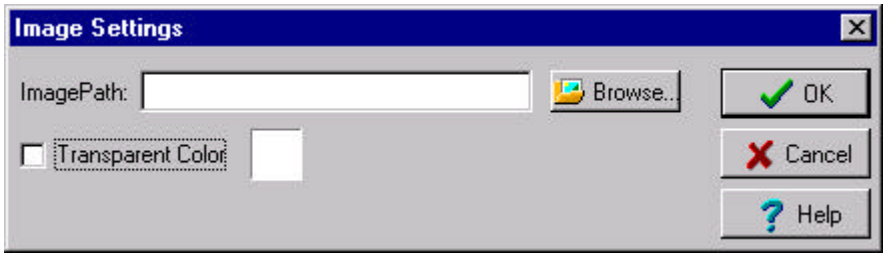

Figure 12.12 shows the image settings dialog, with all the possibilities that are present.

13 Appendix C – INI-files

Set-up

This is not a module INI-file. It is the INI-file for the set-up application.

[GENERAL]
Setup=encrypted password

[PROJECTS]
ProjectsNo=1
Project1=default

Main

The main module has four sections shown below.

[START_UP]
GeneralScreen=1

[TERMINATE]
GeneralScreen=1

[FLOW]
CustomerID=2
Supervisor=encrypted password
Maintenance=encrypted password
Quit=encrypted password

[MODULES]
Idle=idle.dll
IdleSetup=idle.ini
Identify=identify.dll
IdentifySetup=identify.ini
Deposit=deposit.dll
DepositSetup=deposit.ini
Dispense=dispense.dll
DispenseSetup=dispense.ini
Option1=option1.dll
Option1Setup=option1.ini
Option2=option2.dll
Option2Setup=option2.ini
Option3=option3.dll
Option3Setup=option3.ini
Option4=option4.dll
Opotion4Setup=option4.ini
Finish=finish.dll
FinishSetup=finish.ini
Update=update.dll
UpdateSetup=update.ini
Off=off.dll
OffSetup=off.ini
CustomerDB=customerdb.dll
CustomerDBSEtup=customerdb.ini
Receipt=receipt.dll
ReceiptSetup=receipt.ini
Commission=commission.dll
CommissionSetup=commission.ini
Check=check.dll

CheckSetup=check.ini
OnLine=online.dll
OnLineSetup=online.ini
LogFile=logfile.dll
LogFileSetup=logfile.ini
Display=display.dll
DisplaySetup=display.ini

Idle

The idle module has only one section that is shown below.

[FLOW]
Language=1
ReceiptSelection=0

Identify

[CARDNO]
Activated=1
MaxTries=5
OnLineCheck=0
PIN=1
WaitTime=10000

[ACCOUNT1]
Activated=1
MaxTries=5
OnLineCheck=1
PIN=1
WaitTime=10000

[ACCOUNT2]
Activated=0
MaxTries=5
OnLineCheck=1
PIN=1
WaitTime=10000

[TOTALID]
Combinations=4
LinkedAccounts=1
OnLineCheck=1
LocalCheck=1

Check

[CARD]
CDVCheck=1
CDVSum=10
DigitSum=1
CDVFactor=21212-12121-21212-12121
CDVPrefix=00000-00000-00000-00000
FormatString=123PP-PPPPP-PPPPP-PPPPP
FormatChar=14
NoRanges=1
IgnoreLast=3
ExternalCheck=0
ExternalProgram=
Ranges1=11111-11111-11111-11111...11111-11111-11111-11111

[ACCOUNT1]

CDVCheck=1
CDVSum=10
DigitSum=0
CDVFactor=21212-12121-21212-12121
CDVPrefix=00000-00000-00000-00000
FormatString=123TT-TTTTT-TTTTT-TTTTT
FormatChar=18
NoRanges=4
IgnoreLast=4
ExternalCheck=0
ExternalProgram=
Ranges1=00000-10000-10000-10000...10000-10000-10000-10000
Ranges2=20000-10000-10000-10000...30000-10000-10000-10000
Ranges3=50000-10000-10000-10000...99999-99999-99999-99999
Ranges4=55555-55555-55555-55555...99999-99999-99999-99999

[ACCOUNT2]
CDVCheck=1
CDVSum=8
DigitSum=0
CDVFactor=21212-12121-21212-12121
CDVPrefix=11111-11111-11111-11111
FormatString=123OO-OOOOO-OOOOO-OOOOO
FormatChar=13
NoRanges=3
IgnoreLast=8
ExternalCheck=0
ExternalProgram=
Ranges1=00000-10000-10000-10000...10000-10000-10000-10000
Ranges2=20000-10000-10000-10000...30000-10000-10000-10000
Ranges3=50000-10000-10000-10000...99999-99999-99999-99999

Receipt

During the analysis phase I started to read the specification of the Forms Model and Media Description in CWA 13449-3 XFS interface specification Part 3[7]. The script language described in this specification I will put inside a file with the extension FRM. Each file will represent a receipt.
The INI-file will contain paths to each receipt. This is where I ended up with the receipt part.

Display

This is the part that I put the most effort in, most because of my deep interest in graphics. Each slide is represented in this INI-file with a few sections described below.

[SLIDE]
ServiceDisplay=Select Language
RemoteDisplay=Select Language
DisplayTime=0
NoOfTextFields=1
NoOfButtons=1
NoOfImages=1
Enabled=1

[SLIDE_BACKGROUND]
Picture=back.bmp

[SLIDE_TEXTFIELD1]
Text=SELECT LANGUAGE
Top=43

[7] See http://www.cenorm.be/isss/Workshop/XFS/cwa13449/Default.htm

Left=11
Right=89
Bottom=55
HAlign=2
VAlign=1
AutoSize=1
Font=Arial Black
Bold=1
Italic=0
Underline=0
FontSize=32
Color=8388608

[SLIDE_BUTTON1]
Picture=DanishRightButton.BMP
Top=33
Left=70
Right=100
Bottom=41
Transparent=1
TransparentColor=40304837
Text=
HAlign=2
VAlign=1
AutoSize=0
Font=Arial Black
Bold=0
Italic=0
Underline=0
FontSize=13
Color=16711680
Action=9
ButtonID=5

[SLIDE_IMAGE1]
Picture=Button.bmp
Top=37
Left=32
Right=70
Bottom=62
Transparent=1
TransparentColor=37623321