Figure 32 : UML model of Kiebel Car Media

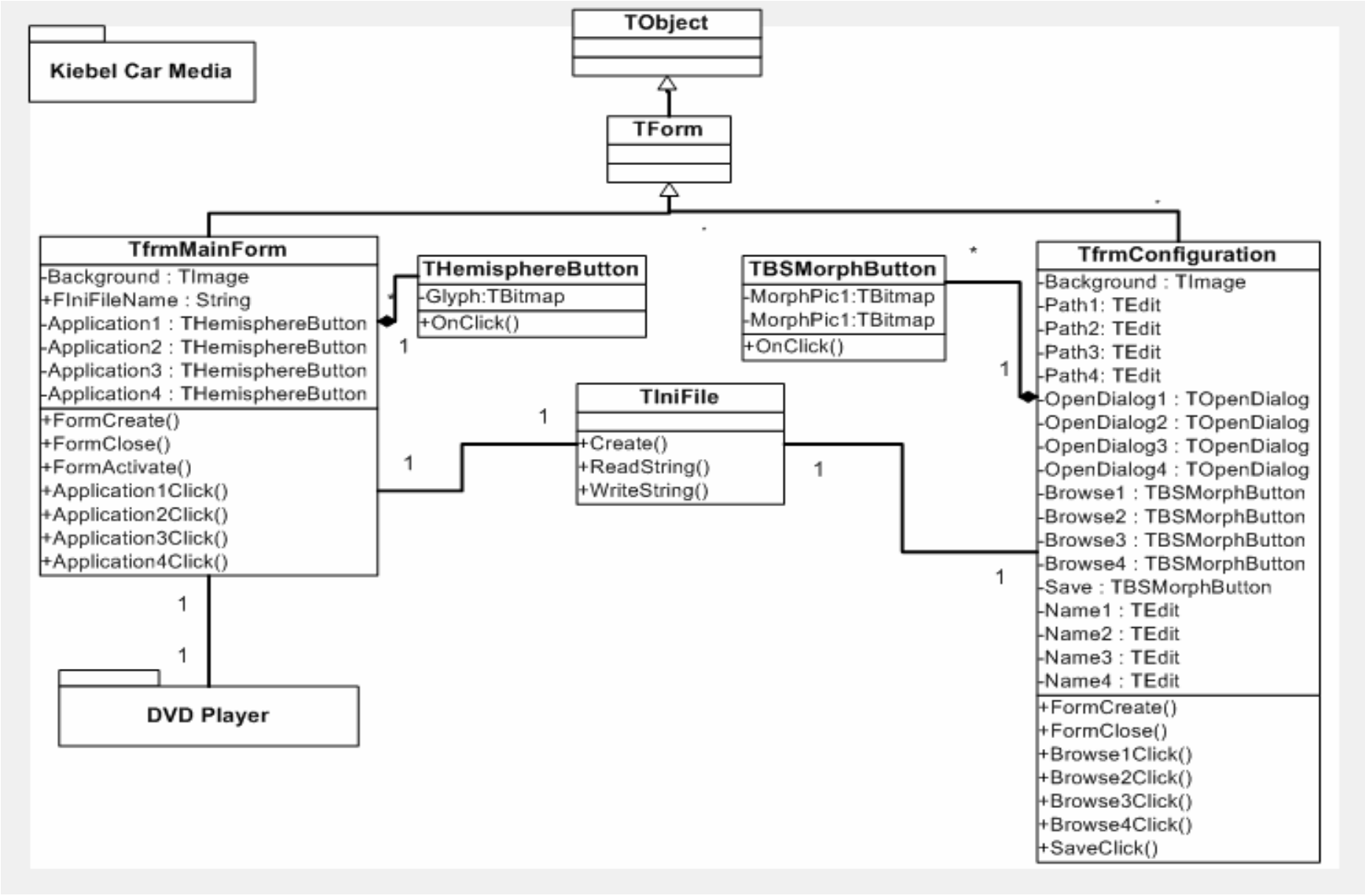
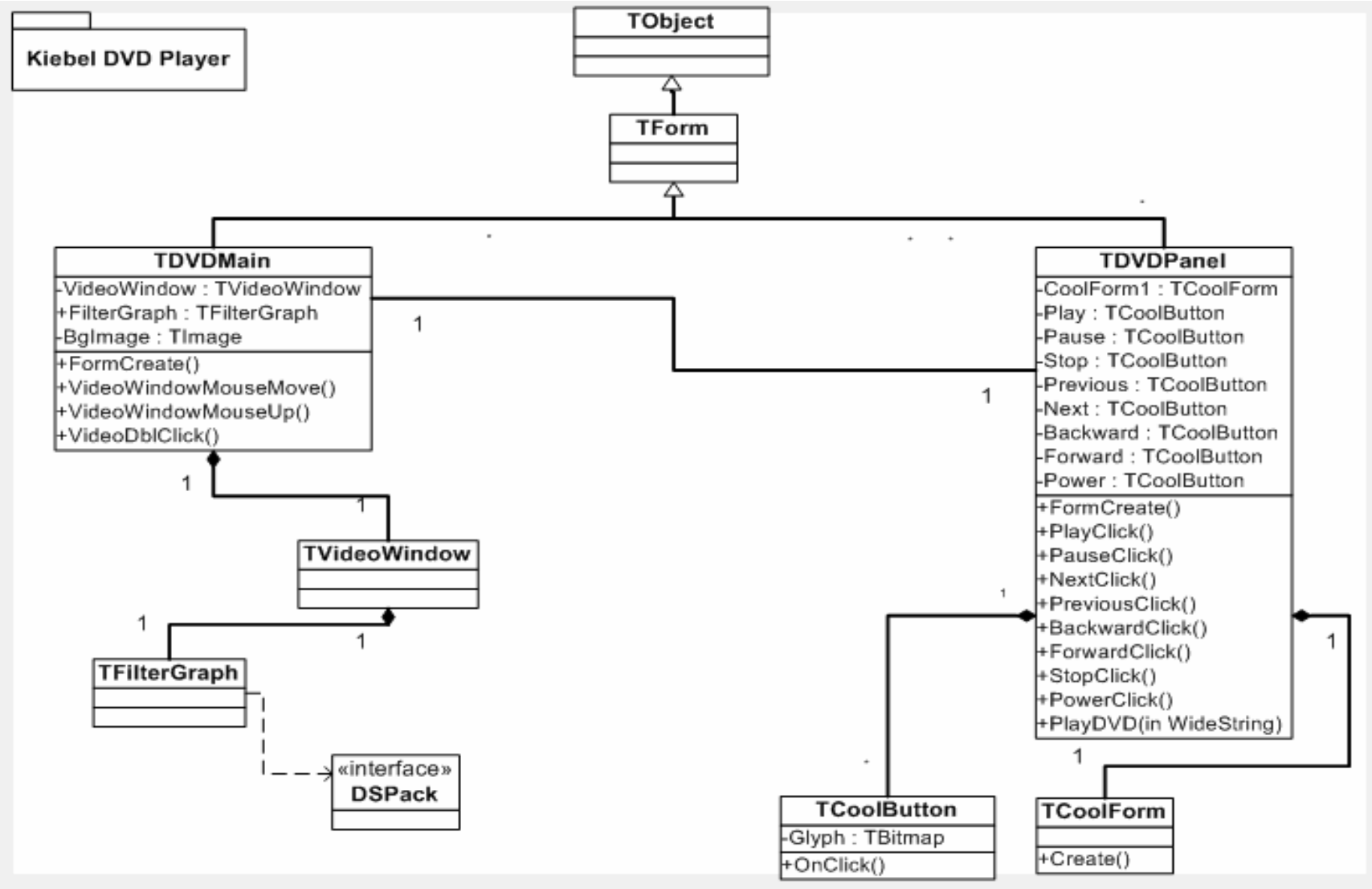


Figure 33 : UML model of Kiebel DVD Player



7.5.3 Conclusion

Finally the high level or the architectural model along with the designs designed in unified modelling language is record in a formal document called '*Design specification document'* or *'Internal Specification Document'*.

This document commonly serves as an input to the developers to proceed with the implementation phase which holds both the programming and the integration process.

7.6 Implementation

In engineering and computer science, an *implementation* is the practical application of a method or algorithm to fulfill a desired purpose. For example, one might create a computer program that sorts a list of numbers in ascending order. To do so, one would *implement* a known method of sorting [21 & 24].

7.6.1 Programming Process

The programming process takes the internal or detail design specification document as input and produces the executable software modules as output. Since most software is tremendously complex, it is decomposed into several largest to manage the complexity. Different teams of people may be assigned to develop different modules. Later these modules will be integration and tested for the release of the product. All these process will be done by the configuration management department which management plays a critical role in the programming of the software modules.

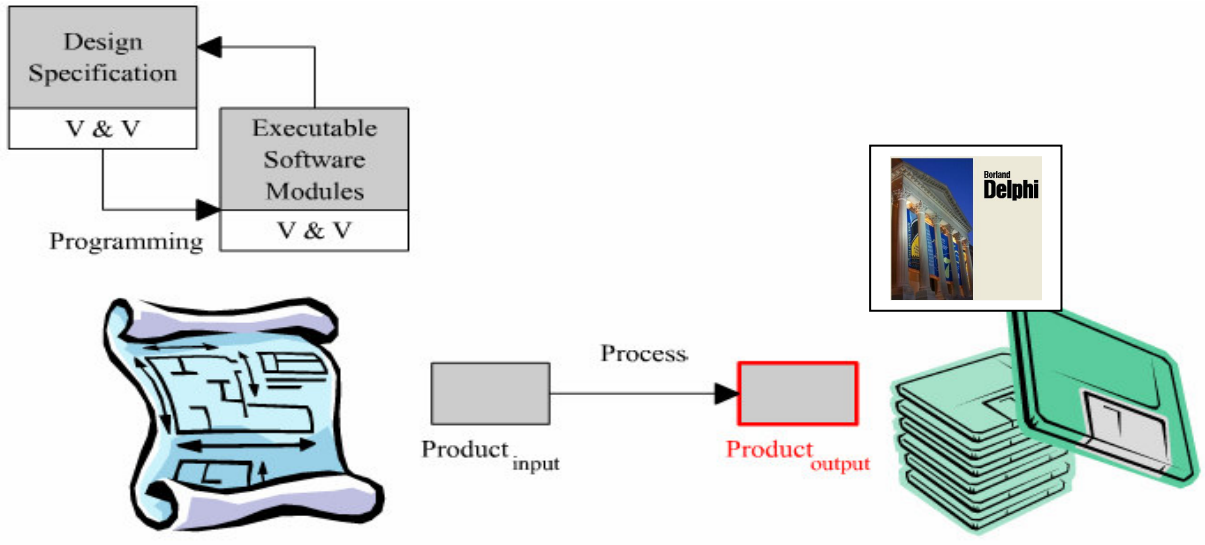


Figure 34: Programming Process

All the programming has been implemented in object-oriented language Delphi 7.0, since this programming language is the company strategy which has been specified in the communicated requirements.

The details about the source code of the Kiebel Car Media and the Kiebel DVD Player are shown in *Appendix A* and *Appendix B* respectively.

7.6.2 Integration Process

The Integration process takes the executable software modules as input and produces the integrated software product as output. Integration testing plays an important role to ensure that the product is integrated successfully.

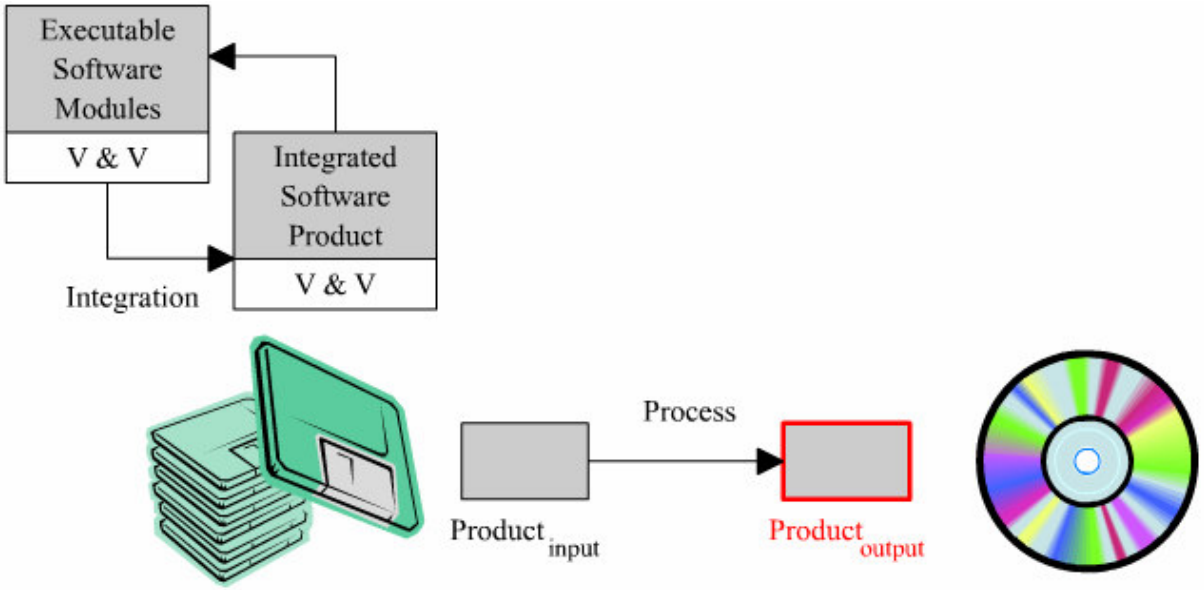


Figure 35: Integration Process

Here to implement the integration process the software *'Install Shield Express'* has been used since it has been specified in the communicated requirements.

This product is very easy and user-friendly to integrate all the executable software modules in to one single executable file. Features like license agreement customized banner or image of the company can also be included using this software. Options like automatically indicating the end user about new updates can be implemented during this process. Various installations like single executable, CD-ROM version or install from the web can also be created for customized needs. Reports can be created as a log file after the build process has been completed, which is very useful while updating the software version.

The life cycle of the software install shield is shown in the below figure.

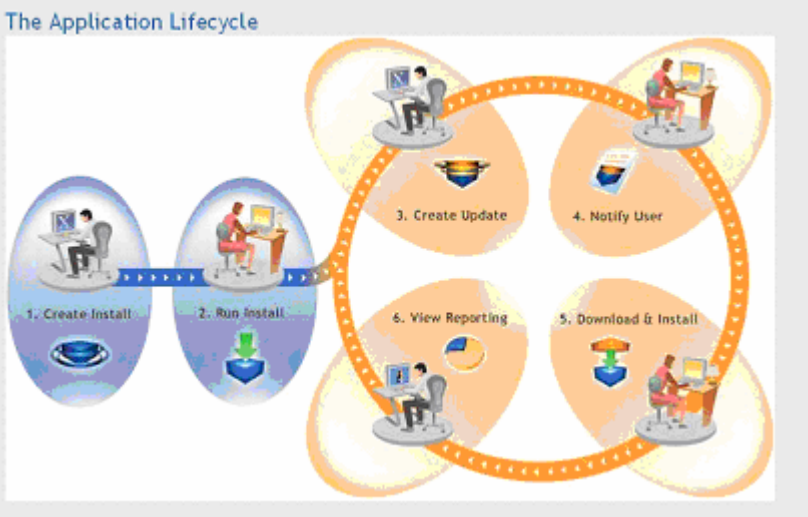


Figure 36: Lifecycle of Install shield software 78

7.7 Testing

7.7.1 Software Testing

Software testing is a process used to identify the *correctness, completeness and quality* of developed computer software. Actually, testing can never establish the correctness of computer software. It can only find defects, not prove that there are none. There are a number of different testing approaches that are used to do this ranging from the most informal *ad hoc* testing, to formally specified and controlled methods such as automated testing. The quality of the application can and normally does vary widely from system to system but some of the common quality attributes include *reliability, stability, portability, maintainability and usability*.

Software testing may be viewed as a sub-field of software quality assurance but typically exists independently (and there may be no SQA areas in some companies). In Software Quality Assurance, software process specialists and auditors take a broader view on software and its development. They examine and change the software engineering process itself to reduce the amount of faults that end up in the code or deliver faster [25].

A problem with software testing is that the number of *defects* in a software product can be very large, and the number of configurations of the product larger still. Bugs that occur infrequently are difficult to find in testing. A rule of thumb is that a system that is expected to function without faults for a certain length of time must have already been tested for at least that length of time. This has severe consequences for projects to write long-lived reliable software. One of the common misunderstandings of software testing is that it is performed by an independent group of testers after finishing the software product and before it is shipped to the customer. But this approach leads to the common effect that the test team is used as project buffer to compensate project delays. Additionally the *earlier a defect* is found the *cheaper it is to fix I* [21]*t*.

7.7.2 Alpha testing

In software development, testing is usually required before release to the general public. In-house developers often test the software in what is known as 'alpha' testing which is often performed under a debugger or with hardware-assisted debugging to catch bugs quickly. It can then be handed over to quality assurance staff for additional testing in an environment similar to how it was intended to be used. This is often known as the second stage of alpha testing [21].

7.7.3 Beta testing

Following that, limited public tests known as beta-versions are often released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta-versions are made available to the open public to increase the feedback field to a maximal number of future users [21].

7.7.4 Gamma testing

There are companies that introduced the so-called gamma tests, which means feature-completed, but the software did not run through all the in-house quality checks. Some cynics refer to software release as "gamma testing" [21].

7.7.5 Software Testing Activities

Some of the testing activities that are commonly used in the software industry to optimise the defects are listed below. Each and every testing activity is for specific purpose of testing [21].

- Unit testing
- Integration testing
- System testing
- Regression testing
- Load testing
- Performance testing
- Stress testing
- Security testing
- Installation testing
- Usability testing
- Stability testing
- Authorization testing
- User acceptance testing & Conformance testing

Our project mainly deals with *'Stress Testing'*, since which has been specified in the communicated test requirements.

7.7.6 Stress Testing

Stress testing is a form of testing which is used to determine the stability of a given system or entity. It often involves testing something beyond its normal operational capacity in order to observe the results. For example, a web server may be stress tested using scripts, bots and various denials of service tools. Medical context: Cardiac stress test is used mostly commonly to identify problems with blood flow to the heart muscle, sometimes other issues.

Generally stress testing is nothing but testing the software under stress that is abnormal operational conditions. This can be done by automation process.

Test Complete 3.0 is a tool used here to automate the testing process.

The below table3 and table 4 shows the test cases of Kiebel Car Media and Kiebel DVD Player respectively.

Actions	Expected Result	Actual Result	Status
In the Configuration Window, Click on SAVE button Without filling the fields in it.	Throws warning message "Please fill the Application Name 1"	Throws warning message "Please fill the Application Name 1"	PASS
Filling out the first Application name and click on SAVE button.	Throws warning message "Please BROWSE the PATH"	Throws warning message "Please BROWSE the PATH"	PASS
In the Configuration Window, Click on SAVE button.	Throws warning message "Please fill the Application Name 2"	Throws warning message "Please fill the Application Name 2"	PASS
Filling out the first Application name 2 and click on SAVE button.	Throws warning message "Please BROWSE the PATH"	Throws warning message "Please BROWSE the PATH"	PASS
In the Configuration Window, Click on SAVE button..	Throws warning message "Please fill the Application Name 3"	Throws warning message "Please fill the Application Name 3"	PASS
Filling out the first Application name 3 and click on SAVE button.	Throws warning message "Please BROWSE the PATH"	Throws warning message "Please BROWSE the PATH"	PASS
In the Configuration Window, Click on SAVE button..	Throws warning message "Please fill the Application Name 4"	Throws warning message "Please fill the Application Name 4"	PASS
Filling out the first Application name 4 and click on SAVE button.	Throws warning message "Please BROWSE the PATH"	Throws warning message "Please BROWSE the PATH"	PASS
In the Configuration Window, Click on SAVE button..	Windows closes and creates an INI file in the working directory.	Windows closes and creates an INI file in the working directory.	PASS

Actions	Expected Result	Actual Result	Status
In the Main Window, Click on the Application button 1.	Launch the Application corresponding to the PATH in INI file	Launch the Application corresponding to the PATH in INI file	PASS
In the Main Window, Click on the Application button 2.	Launch the Application corresponding to the PATH in INI file	Launch the Application corresponding to the PATH in INI file	PASS
In the Main Window, Click on the Application button 3.	Launch the Application corresponding to the PATH in INI file	Launch the Application corresponding to the PATH in INI file	PASS
In the Main Window, Click on the Application button 3.	Launch the Application corresponding to the PATH in INI file	Launch the Application corresponding to the PATH in INI file	PASS

Table 3 : Test Cases for Kiebel Car Media

Actions	Expected Result	Actual Result	Status
Insert a CD-ROM in the drive and click on PLAY.	Throws Warning message "NO DVD Volume found"	Throws Warning message "NO DVD Volume found"	PASS
Insert a DVD in the drive and click on PLAY.	Plays the Title or Root Menu.	Plays the Title or Root Menu.	PASS
Move the move over the menus shows in the root menu.	Highlights the menu selected.	Highlights the menu selected.	PASS
Click on the high-lighted menu.	Play the respective menu.	Play the respective menu.	PASS
Click on PAUSE Button	Pause the movie	Pause the movie	PASS
Click on PLAY Button	Plays the movie	Plays the movie	PASS
Click on Forward Button	Forwards the movie	Forwards the movie	PASS
Click on Rewind Button	Rewinds the movie	Rewinds the movie	PASS
Click on Next Button	Plays the next Track	Plays the next Track	PASS
Click on Previous Button	Plays the Previous Track	Plays the Previous Track	PASS

Click on STOP Button	Stops the movie and return to ROOT menu	Stops the movie and return to ROOT menu	PASS
Move the mouse over every button	Highlights the menu under the mouse	Highlights the menu under the mouse	PASS
Click on Power Button	Will close the application.	Will close the application.	PASS

7.7.7 Test Automation

The use of software to control the execution of tests, the comparison of actual outcomes to predicted outcomes, the setting up of test preconditions, and other test control and test reporting functions.The use of technology, such as capture/playback and data comparison, to enable test scripts/cases to be developed and executed (potentially in an unattended or off-hours mode).

'TestComplete 3' is the automation tool here to test the developed Delphi executable files. AutomatedQA the company which introduced Test Complete has repeatedly shocked the Quality Assurance world by offering a full-featured testing and debugging solutions for prices that are fractions of the cost of "the big names TestComplete 3 adds HTTP load, stress and scalability testing, distributed testing and improved unit testing to its already formidable list of features.

As the name implies, TestComplete is AutomatedQA's complete application testing system offering *automated functional, unit, regression, distributed and HTTP performance testing* in one easy to use and totally integrated package. TestComplete allows you and your team to implement comprehensive software testing strategies, "automating the non-automatable" for maximum ROI (return on investment).

The recorded test scripts based on the created test cases produce the result and its causes if fails. This result can be viewed in a browser or inside Test Complete 3.0. A sample result log is shown in the figure 37.

The recorded test scripts can be found in *Appendix C*. Also there is a movie file which shows how the testing process has been done from the recorded scripts. The movie file *'Kiebel Car Media Testing'* can be found in the attached CD-ROM.

Test Log

N		Name	Start Date	Start Time	End Date	End Time	Run Time	Computer
ok	13	Main	30.09.2004	17:36:26	30.09.2004	17:36:26	0:00:00	QUEEN
x	14	Test1	30.09.2004	17:36:30	30.09.2004	17:37:01	0:00:31	QUEEN
x	15	Test1	30.09.2004	17:42:18	30.09.2004	17:42:59	0:00:41	QUEEN
x	16	Test1	30.09.2004	17:43:48	30.09.2004	17:44:54	0:01:06	QUEEN
ok	17	Test1	30.09.2004	17:45:23	30.09.2004	17:45:42	0:00:19	QUEEN
x	18	Test1	30.09.2004	17:48:34	30.09.2004	17:49:11	0:00:37	QUEEN
x	19	Test1	30.09.2004	17:52:14	30.09.2004	17:52:19	0:00:05	QUEEN
ok	20	Test1	30.09.2004	17:52:27	30.09.2004	17:52:53	0:00:26	QUEEN
ok	21	Test1	30.09.2004	17:57:18	30.09.2004	17:59:27	0:02:09	QUEEN
x	22	Test1	30.09.2004	18:03:01	30.09.2004	18:05:52	0:02:51	QUEEN
x	23	Test1	30.09.2004	18:06:46	30.09.2004	18:08:17	0:01:31	QUEEN
ok	24	Main	30.09.2004	18:09:20	30.09.2004	18:09:20	0:00:00	QUEEN
x	25	Test1	30.09.2004	18:09:34	30.09.2004	18:11:08	0:01:34	QUEEN
ok	26	Test1	30.09.2004	18:11:54	30.09.2004	18:13:16	0:01:22	QUEEN
ok	27	Test1	30.09.2004	18:15:28	30.09.2004	18:17:59	0:02:31	QUEEN

C:\Exe Files for testing\Kiebel Car Media\Results\Kiebel Car Media.htm - Microsoft Internet Explorer

Datei Bearbeiten Ansicht Favoriten Extras ?

Zurück Suchen Favoriten Medien

Adresse C:\Exe Files for testing\Kiebel Car Media\Results\Kiebel Car Media.htm

Google Web-Suche PageRank 139 blockiert Optionen

SEARCH TopSearches Dating Shopping Games Travel Practical

err	16	Test1	30.09.2004	17:43:48	30.09.2004	17:44:54	0:01:06
ok	17	Test1	30.09.2004	17:45:23	30.09.2004	17:45:42	0:00:19
err	18	Test1	30.09.2004	17:48:34	30.09.2004	17:49:11	0:00:37
err	19	Test1	30.09.2004	17:52:14	30.09.2004	17:52:19	0:00:05
ok	20	Test1	30.09.2004	17:52:27	30.09.2004	17:52:53	0:00:26
ok	21	Test1	30.09.2004	17:57:18	30.09.2004	17:59:27	0:02:09
err	22	Test1	30.09.2004	18:03:01	30.09.2004	18:05:52	0:02:51
err	23	Test1	30.09.2004	18:06:46	30.09.2004	18:08:17	0:01:31
ok	24	Main	30.09.2004	18:09:20	30.09.2004	18:09:20	0:00:00
err	25	Test1	30.09.2004	18:09:34	30.09.2004	18:11:08	0:01:34
ok	26	Test1	30.09.2004	18:11:54	30.09.2004	18:13:16	0:01:22
ok	27	Test1	30.09.2004	18:15:28	30.09.2004	18:17:59	0:02:31

Figure 37: Test log from TestComplete 3.0

7.8 Delivery & Maintenance Process

The delivery process takes the integrated software product as input and produces the delivered software product as output. Acceptance testing is commonly conducted by the client to ensure that the delivered product meets the requirements specified in the contract. Once the client accepts the delivery, the software product is installed and used.

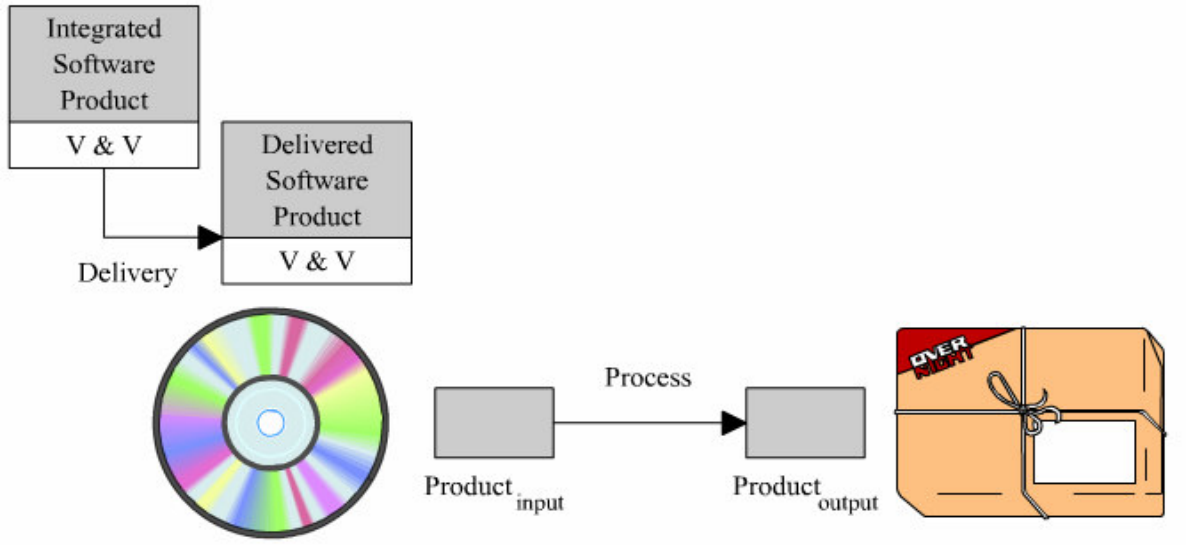


Figure 38: Delivery and Maintenance Process -1

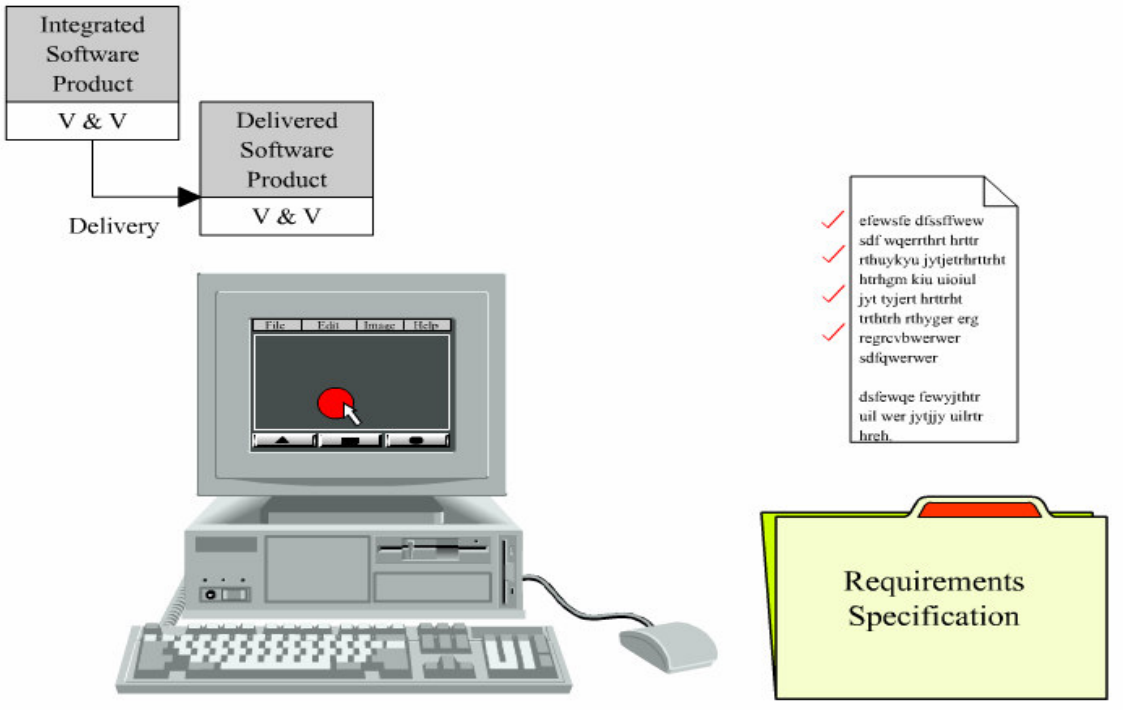
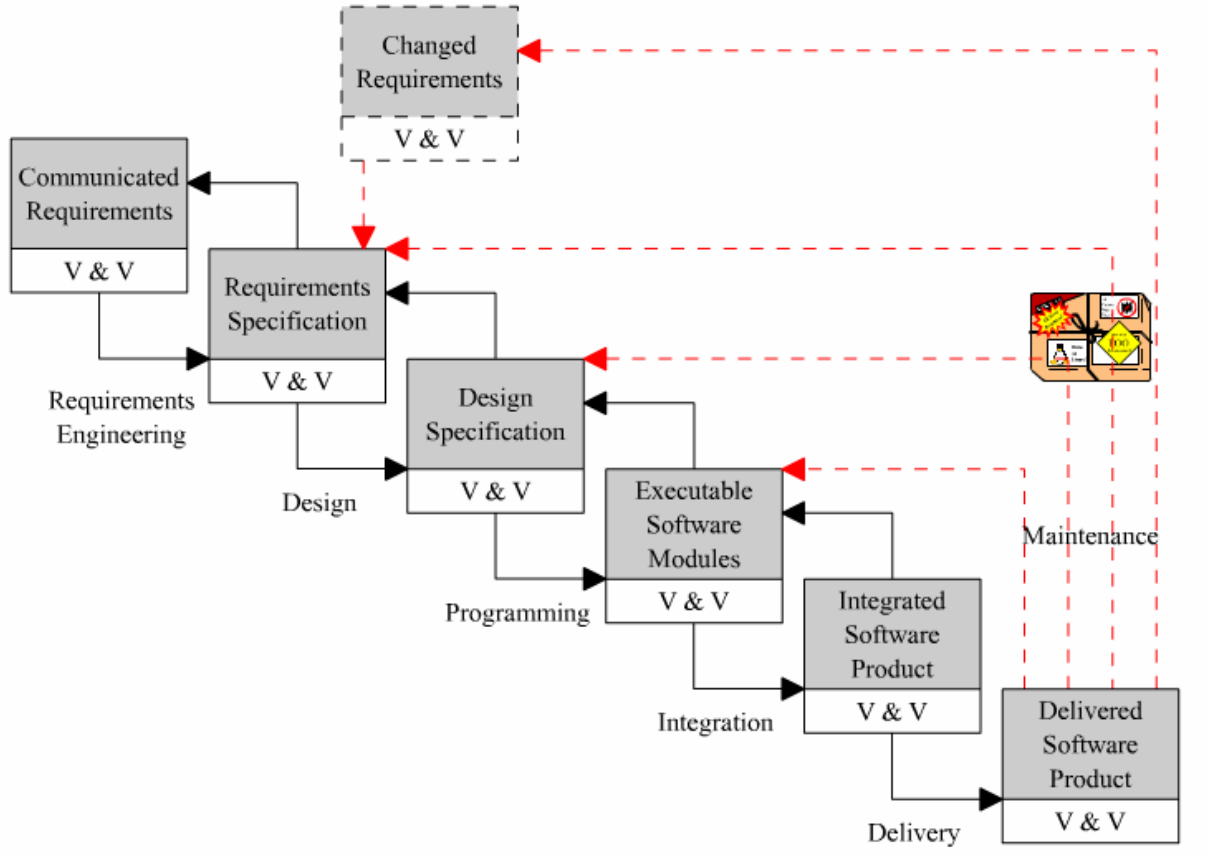


Figure 39: Delivery and Maintenance Process-2

Every successful software product goes to an evolution which requires changes to the software product after its delivery. This evolution is commonly represented by a software version number. Changes may be required for variety of reasons such as fixing identified bugs, providing new functionality, adapting the software to new technologies, or porting the software to new platforms. These changes are called 'Software Maintenance' represented by the dashed areas in the life cycle figure. Software Maintenance resolves in the modification of the software in different life cycle products from changed requirements to executable software modules.

Figure 40: Delivery and Maintenance Process-3



Chapter 8
Conclusion

By the end of my master thesis, the development of the whole media entertainment system along with DVD Player software has been developed. Several testing process were done to ensure the reliability of the software, 95% of the project testing were successful and the remaining 5% of the tests has to be done on the hardware.

The Graphical User Interface prototype acts as a good bridge between the developer and the client. It proves its user friendliness and it was very helpful for further improvements in requirements.

The successful architectural designs were helpful to finish the development in time which made the project life cycle shorter.

The project is extendable with various other functionalities with the help of the database in future due to its flexible architecture.

Generally the software proves its usability with a well customized user interface and plays as a low cost solution for all automobile customers.

Appendix A

Delphi Source code of Kiebel Car Media

Since the source code is too big to include in this document, the source code has been included in the attached CD-ROM. Please refer the CD-Browser demo or index.html to download the source codes.

Appendix B

Delphi Source code of Kiebel DVD Player

Since the source code is too big to include in this document, the source code has been included in the attached CD-ROM. Please refer the CD-Browser demo or index.html to download the source codes.

Appendix C

Test Complete Delphi Test Script

Since the recorded test script code is too big to include in this document, the source code has been included in the attached CD-ROM. Please refer the CD-Browser demo or index.html to download the test scripts.

References

[Can03] Marco Canto. *Mastering Delphi 7.0,* Sybex Inc., USA, 2003.

[1] http://tekgems.com/Products/creative-multimedia-remote-cimr100.htm

[2] http://www.xenarc.com/product/cp1000.html

[3] http://www.thisstrife.com/carproject/

[4] http://www.pricepc.com/html/carpc.php

[5] http://www.mp3car.com/

[6] http://www.letscommunicate.co.uk/

[7] http://home.debitel.net/user/incar-media/incar/in-car-auto-multimedia.htm

[8] http://www.delphipraxis.net/

[9] http://homepages.borland.com/torry/

[10] http://www.torry.net/

[11] http://www.delphipages.com/

[12] http://www.csd.net/~cgadd/delphi.htm

[13] http://www.greatis.com/delphi/

[14] http://www.swissdelphicenter.ch/torry/

[15] http://delphi.helli.de/

[16] http://www.paranoia.clara.net/

[17] http://www.geocities.com/~franzglaser/tp.html

[18] http://www.andrew.cmu.edu/course/46-939/notes/SoftLifeC.PDF

[19] http://www.dacs.dtic.mil/techs/prototype/DACS-Prototype.pdf

[20] http://en.wikipedia.org/wiki/Software_development_process

[21] http://www.nationmaster.com/encyclopedia/

[22] http://courses.cs.vt.edu/~csonline/SE/Lessons/Waterfall/

[23] http://c2.com/cgi/wiki?WaterfallModel

[24] http://en.wikipedia.org/wiki/Implementation"

[25] http://www.nationmaster.com/encyclopedia/Software-testing

[26] http://www.automatedqa.com

[27] http://www.udsl.com/