FYP Final Report 2010

Student Teacher PDF Assignment and Assessment Tool

Colin O' Shea - 0603104

Supervisor – J.J. Collins
4/19/2010

Abstract

As computers become more prevalent in work, education and life in general the scope of this project is to measure the functionality and capabilities of using touch screen technology in an educational setting. One of key concepts of the project is that the user will only have to use a tablet pc stylus for input - thus making it a truly touch screen application.

This project's basic concept is to create a software application which will be used by both students and teachers to answer and correct worksheets only with a stylus pen. This will also involve using server side and internet technologies to make the worksheet available for downloading for answering and uploading for correction. This will facilitate never having to leave the touch screen environment – a common flaw in many of touch screen applications.

The other key feature is that the worksheet will be in PDF format. PDF format is the most prevalent form of document type on the internet today. PDF technology has a wide variety of tools that make using it as a worksheet beneficial. PDF already support active form fields, radio buttons and JavaScript. This proof of concept is to add ink input to an already created PDF document.

Acknowledgements

Much thanks to the supervisor whose help was greatly appreciated. His input into planning and design concepts were highly beneficial in the outcome of the project. Knowledge is power and he has it!

Much thanks to my family who listened to the late night phone calls when sanity was lost. The food supply chain was always appreciated in difficult times

“Much love” to [nhc] – Proxykillah and [nhc]-Biosed, whose contribution to the project cannot be understated.

Last but not least, many thanks to Mr.Hayes and Mr.Moroney. Four years of fun.

Table of Contents

Table of Figures

1. Introduction

1.1 Summary

This project is a proof of concept on the functionality of touch screen computing in the educational environment. With the advent of touch screen mobile phones and touch screen user displays people are becoming more used to working or dealing with touch screen technology.

This project is about trying to prove the usefulness of touch screen computing in the classroom. The key concepts of the project are:

- Create an application which only uses tablet stylus as input
- The software will allow for annotation of a PDF document by the student
- The student will then upload document to a server for correction by teacher
- The teacher will download the answered worksheet for correction
- The teacher will correct the answered worksheet and register the mark on a simple marking scheme

When created this leads to a true paperless environment for both the teacher and student. By using the stylus for writing the answers the student does not lose any of the associated benefits of actually writing the answer.

Feedback from the result of a test is one of the most important roles of the teacher (Price, 1997). Red ink annotation onto a paper copy is the traditional solution. It is only in a test setting that a teacher will learn the level of the student's knowledge of the subject. By being able to annotate the worksheet that the student has answered the learning capacity of the student is increased.

This application removes the need for paper and pen altogether. By creating an electronic document instead of a physical document storage system must change. The application will be connected to a server so the student can upload their completed worksheet for the teacher to correct. This is beneficial because it allows the work to be carried out irrespective of geographical location thereby lending itself to the idea of distance learning. Incorporating a server also lessens the chance of loss of papers or even perhaps the common eventuality of a dog eating a student's homework!

Administration tasks are greatly reduced by the concept of a paperless environment. While the success of a true paperless environment for assessment is still unclear (Apperley, 2007) continuing research is ongoing to finalize proper work flow principles to make it a functional replacement for pen and paper.

1.2 Research Objectives

The objectives of this project are:

- Create an application which will only require the use of stylus pen as input
- Encapsulate the entire process of answering a worksheet, marking a worksheet, delivering a worksheet and maintaining scores achieved into one application
- Utilize PDF format as the basis for the electronic document. This means enabling annotations through stylus pen and saving those annotations into the document.
- Measuring the effectiveness of the application through a series of user based tests. Gather test data and propose changes to the system based on data gathered by these tests.

1.3 Research Methodologies

Research methodologies started with an in depth look at competitors to the proposed application. Here is a summary of what i found. I also did a comprehensive search into recent papers regarding touch screen computing in an educational environment.

1.3.1 Competing Products

There are many products already in the marketplace which cater for the annotation of PDF documents. However, only one product is free. None of these products contain a marking and are more driven towards annotation of notes and PDF.

Microsoft "OneNote" (Microsoft OneNote) is Microsoft's tool for note taking and storage. To annotate PDF's it adds the PDF into the OneNote system and then the user can add comments. It is impossible to add to the PDF otherwise. This however is to be accepted as the core of OneNote's application is to be a general note taking system.

Grahl's "PDF Annotator" (PDFAnnotator) is another application which allows for annotation of PDF files. This product allows for the annotation to the PDF file itself.

Evernote's "Evernote" (Evernote) is similar to OneNote in that it is more of a note taking system than a PDF annotator. It takes the PDF and adds it into its own file structure.

Bluebeam's "PDF Revu" (Bluebeam) is another PDF annotator. However this application does not allow for direct writing onto the PDF. It uses a panel to translate the hand written input into text and then transfers the text onto the PDF.

Jarnal's "Jarnal" (Jarnal) is an open source application which allows for PDF annotation. Similar to Evernote it is a note taking software and takes in the PDF and makes it part of its file structure.

Adobe Acrobat 9 (Adobe) is the main software used by corporations to create and manipulate PDF documents. While it does allow for annotations they are entered in text format and it does not allow for input from a stylus pen.

Foxit's Foxit Reader (Foxit)is the main competitor to Adobe Acrobat. Similar to Acrobat it allows for annotation of documents but only in text format. No input from stylus is allowed.

Apart from Jarnal (Jarnal) all these product cost a significant amount of money. Also none of these could be used to keep a marking system. None of these products allow for transfer of the files directly within their system.

What makes my application different is the development of a marking scheme and an all in one solution for answering and assessing worksheets.

1.3.2 Research Papers

The investigation of touch screen in the educational environment has truely evolved over the last three to five years . in England an extensive research into the use of touch screen was carried out in 2008 by Elsevier (Panagiotis Siozos, 2008). It based it research on identifing the key challenges faced by the computer based asseesment model.

They built a prototype and had successful trials in an English educational setting. Teachers showed a real interest in changing existing pedagogies towards a more high tech way. Teachers also appricieated the idea of quicker turn around time in the scoring and analysis of assessment (Sim, 2004).

Cheryl L. Willis (Willis) stated the use of tablets strengthen a students capacity to learn and interact with others. She also advocated the use of speech recognition and that further investigation should be done in to the subject.

Kenrick Mock (Mock) describes the tablet pc as an "effective tool for grading". He also found that it benifitted him greatly when in the classroom. The ability to write on PowerPoint presentations with a pen was of great use to both him and students. He also noted that for a teacher to be effective with the tablet pc some training in both the software and hardware involved would be needed. Wei-Xiang (Xiang) had very similar findings in his review of the subject.

I am interested to find out whether it will be of benefit to an irish educational setting. Cultural diversity can negate the findings of a study placed in a different region. My hope was to prove that providing an all inclusive program for both teacher and student in the realm of Computer based assesment could be both a viable product and successful in an Irish educational setting

1.4 Overview of Report

In the opening chapter I plan to give a brief summary of my project, as well as what I hope to investigate from researching this project.

 In the second chapter I am going to outline the main research topics that I have chosen and what I learned during my research of these topics.

In the third chapter I plan on discussing different design and Implementation techniques.

In the forth chapter I shall cover the evaluation and testing done on the final product.

In the fifth and final chapter I shall give my conclusions on this project and any relevant points I have to make.

2. Background Research

2.1 PDF File Formats

2.1.1 Structure of a PDF Document

To understand how to work with PDF files I needed to learn about the format of the files and how they are created. This meant reading "PDF Reference Version 1.7" from Adobe Systems.

2.1.2 What is a PDF?

Printable Document Format is an extensible page description protocol that implements the native file format of Adobe Acrobat products. The goal of the format was to provide an independent capability for the interchange of high resolution documents. These documents can contain text, graphics, multimedia elements or possibly URLs.

The format also supports text search, random access of data, bookmarks, and interactive page elements such as checkboxes and text edit fields. Finally latest version also contains the capability of encryption, compression and JavaScript.

- PDF – created by John Warlock of Adobe Systems in 1984.
- PDF has gone through seven versions. Current version is 1.7.

The PDF combines three technologies. One of the best descriptions I found was from Wikipedia (wikipedia):

- A subset of the PostScript page description programming language, for generating the layout and graphics.
- A font-embedding/replacement system to allow fonts to travel with the documents.
- A structured storage system to bundle these elements and any associated content into a single file, with data compression where appropriate.

The structure of a PDF file is based on the concept of objects and also layers.

From "PDF Reference Version 1.7" (Adobe System Incorporated, 2006)

"A *PDF document* consists of a collection of *objects* that together describe the appearance of one or more *pages*, possibly accompanied by additional interactive elements and higher-level application data."

The concept of layers comes from other Adobe products such as Photoshop. When Adobe created Version 1.4 they introduced the concept of layers. This is where if a new “object” needs to be added to the PDF a transparent layer is placed on the original and the two objects are composited together with the transparency of the objects still upheld. Raster images are used if it’s a picture.

2.1.3 Parts of a PDF File

A PDF is made of the following components. – From “PDF Reference Version 1.7” – Chapter 3.

2.1.3.1 Objects

A PDF document is a data structure composed from a small set of basic types of data objects.

- Boolean values
- Integer and real numbers
- Strings
- Names
- Arrays
- Dictionaries
- Streams
- The null object

Objects may be labeled so that they can be referred to by other objects. A labeled object is called an *indirect object*.

2.1.4 File Structure

The file structure can be divided into four sections (Figure 2). Each will be discussed here.

2.1.4.1 Header

The header is a one line statement. The first line of a PDF file specifies the version type with which the PDF adheres to. This will be written in a PostScript style comment.

It will appear as follows -

%PDF-1.7

All text appearing after the % sign until the end of the line will be disregarded by the processor of the file.

2.1.4.2 Body

The body of the PDF document consists of the objects that make up the documents contents. These may be text, image data, fonts or annotations for example. The body may also contain objects which are invisible such as security features and logical structure.

2.1.4.3 Cross-Reference Table

The cross reference (Figure 3) table of a PDF is simply put a map of directional layout for a PDF. For each object of the PDF there is a line which is 20 bytes long; which includes the line end characters. The first ten bytes of the line describe the object's offset - where it is positioned in the PDF. The next six characters contain a space and a five digit number which is the object's "generation number". This number is similar to a unique identifier. The final few bytes contain an indicator to tell whether the object is free or in use. The final element of the line is the end of line marker.

2.1.4.4 Trailer

The PDF trailer enables the application viewing the PDF to locate and reference the cross reference table and certain special objects. The very last line of a PDF file contains only the end-of-file marker "%EOF". The trailer contains what is called the "trailer dictionary". This gives the processor of the file all the relevant details pertaining to the PDF. It contains a reference to the size or the total number of entries in all sections of the document's cross reference table. The trailer also contains the "/Root" key. The Root key is a reference to the "catalog" object. The "catalog" object is a listing of all the objects contained in the documents. Finally there is the "/Info" key. This is optional and can contain extra information regarding the document structure.

2.1.5 Document Structure

The PDF document structure specifies how the basic object types are used to represent components of a PDF document: pages, fonts, annotations, and so forth.

2.1.6 Content Streams

A PDF *content stream* contains a sequence of instructions describing the appearance of a page or other graphical entity. The stream is similar to a string in programming terms - a length of text. However unlike strings a stream can be broken into smaller pieces. This is common practice due to the size of streams. Streams are packaged in a particular way, so they can be located quickly by the processor of the file. This packaging is called an indirect object.

2.1.7 Indirect Objects

An indirect object is a number object. The content can be native types of PDF object (Boolean, number etc). The content of the object must be bracketed between the "obj" and "endobj" keywords in the code for the PDF file.

The advantage of using objects is each object in the file is referenced within the cross reference table. This means they can be reused by any pages or dictionaries within the document.

2.1.8 The Catalog Tree

The catalog is a dictionary containing the root node of a PDF document. The Catalog has an entry for each page in the /Pages (this is the document's page tree).

2.1.9 The Pages Tree

The pages tree is similar to the catalog tree. The page contains all the information regarding that page. The leaves in that page are references to the objects which are used in that page.

2.2 Touch Screen Technology

In this section I shall discuss touch screen technology. I shall give a brief overview of the history and current hardware technologies being implemented.

2.2.1 History of Touch Screen

History of touch screen technology goes back further than one might think. Research into computers interaction through a touch screen dates back to the 1940's. Patents for the idea go back even further. In 1888 Elisha Grey patented the idea of using an electrical stylus device for the capture of handwriting. In the 1950's Tom Diamond invented the "Styalator" electronic tablet with pen for computer input. He also created software which was for handwriting recognition.

The first real touch screen interface came about in 1975 as part of the PLATO project. (PLATO)

A major step forward in the realm of touch screen computing was in 1985 with the development of Pencept. Pencept was the first entry to the consumer market using tablet and pen input with hand writing recognition instead of the traditional keyboard and mouse.

It is clear that Pencept developers could foresee the role touch screen computing would play in the future. The operating system chosen to run the software was Microsoft's MS - DOS. This association with the cutting edge of touch screen and handwriting technology has continued until the present day.

Research and development of the interface continued throughout the late 1970's and throughout the entire 1980's. Multiple systems of different architectural types were made during this time. By the 1980's many public interfaces such as bank machines and entries systems to buildings would contain simple robust touch screen technologies.

It was not until the early 1990's that mainstream computer manufacturers started to incorporate touch screen technology into home user equipment.

As is common with computer technology Apple were first to bring a successful functioning PDA/tablet to the market with the arrival of the "Apple Newton". It came to market during Apple's dark years and is often used as one of the reasons that Apple nearly went out of business. There was no demand for such a product at the time as the potential end users had to be running Mac OS on their home or office machines to make proper use of it. This meant the business world and the main market of PDA type tools would never purchase it due to the dominance of the Microsoft Operating systems in the work environment.

Two of the current day market leaders of touch screen technology released their first commercial products in the early 1990's. Fujitsu made their first tablet with touch screen in 1993. It was called the "Poqet PC". IBM also released their first of the famous "Thinkpad" series in 1993.

2.2.2 Current Developments and Status of Touch Screen

Currently there is an enormous increase in the market of low cost affordable touch screen computers. With what is now a twenty year history, well established physical implementations of touch screen technology that is cost effective is now easily attainable. Major laptop manufacturers "Asus" and "Acer" are famous for the "netbook". The netbook is a small low cost laptop with low powered processor, commonly the Intel Atom processor. This year these two companies are bringing out touch screen netbooks. This has forced the hand of the major touch screen producers to follow Moore's Law and drop their prices.

A relatively new product to start appearing on the market is the multi touch laptop. As is also the trend these days all major manufacturers have some form of multi touch laptop for sale. It is probable to have a future where all laptops being sold have multi touch screens.

2.2.3 Touch Screen Types

There are two main types of touch screen, active and passive. I will discuss these briefly while also discussing a third type which is common in hand held devices.

2.2.3.1 Passive Touch Screen

Passive touch screens are more common in situations where the human hand will be used rather than a stylus for input. This means that there is no need for a special type of stylus for input . it common to use this style in interfaces where the hand will be used to operate the computer. It contains two resistive sheets of plastic with an air pocket between them. There is a small electrical current running throught the two seperate screens. When contact is made with the screen the outer sheet makes contact with the inner sheet. One sheet corresponds to the y axis . The other sheet corresponds to the x axis. By combining the two measurements provided by the seperate sheets an accurate location can be calculated from the touchscreen.

The issue with this type of technology for the use of writing tablets is that it is impossible to verify which is the point of contact wanted if there is more than one point of contact. Since the screen could be touched in more than one place it becomes hard to know where the user is actually trying to point to. High end machines which use passive technology such as Fujitsu's P1630 have software with "palm resistant " technology. The idea here is to eliminate the area of contact with the biggest circumference from the touchscreen register and only use the area with the small circumference. It is very effective due to the area of the tip of a stylus being much smaller than the palm of a hand.

2.2.3.2 Active Touch Screen

Active touchscreens cannot be activated by the human hand. They rely on input from an "active digitiser" (mobile) . This is the stylus with an eltromagnetic magnet in it. When the stylus comes close to the screen it causes the tablet screen to pick up on the interference created by the magnet. This is how it is possible to hover over these types of screens with the stylus and make the cursor on the screen move. This method is the preferred option for tablet pc due to the single point of contact with the screen that is registered. Resting the palm of your hand on this screen while writing has no effect on the cursor.

2.2.3.3 Projected Capacitance Touch Screen

Although not used in this project I shall briefly describe this form of touch screen. This form is the most popular on hand held devices due to its capability to be used in conjunction with glass. A grid pattern of electrodes is placed under the durable surface of glass. This creates an electrostatic field underneath the surface glass. When the human finger - which has conductivity properties comes in contact with the screen it creates a charge on the grid, thereby locating the point of contact. The grid also allows for multi touch capabilities due to identifiable separate locations of contact.

2.2.4 Component Middleware Platforms Comparison

Definition: *An application server, or appserver, is software that typically interfaces one or more databases to convey processed data to and from a user interface* (bestpricecomputers).

When we hear the word server we automatically think of a machine. The term application server actually refers to a piece of software. The scope of the software's utilization is to be the "business layer" of an n tier system. The business logic of the system is essentially the functions that the software performs on the data held in the EIS. For my project i had to decide which application server best suited my needs

2.2.4.1 Microsoft .NET Framework V Java J2EE

The .NET framework is one of the main competitors to the Java J2EE application server. It is similar to J2EE in that is sits on top of the operating system like the java virtual machine. The major difference between the two is that the java virtual machine only supports the java language whereas the .net framework supports many languages such as c#, VB.NET and ASP.NET.

The .NET framework exists on top of various Microsoft operating systems. With the development of MONO by Novell there is an implementation that will run on *nix operating systems as well.

2.2.4.2 Structure of the .NET framework

2.2.4.2.1 Common Language Runtime (CLR)

This is the core of the framework. This is Microsoft's implementation of the Common Language Infrastructure (.net). In the CLI code is a form of byte code known as the Common Intermediate Language. Developers write their application in the various .NET languages. At compile time the .NET compiler converts the code into CIL code. Then at runtime the CLR's just-in-time complier converts the CIL code into code native to the operating system. This means that the developer does not have to worry about giving processor specific requests because the CIL compiled does this work.

2.2.4.2.2 Common Type System

The CTS gives the .NET framework a capability to define the possible data types and programming constructs supported by the CLR. It defines how they can interact with each other. This means that instances of type can be exchanged between programs written in any of the .NET languages.

2.2.4.2.3 Base Class Library

The BCL is a library of common features available to all languages which use the .NET framework. It is an encapsulation of common functions for writing programs. In the System namespace base types such as string, date and Boolean are held. Other namespaces include System.Collections (lists, queues, stack, and hashtables), System.IO (dealing with file handling), System.NET (network protocols, HTTP, FTP and SSL), System.Runtime (for use with distributed systems, serializing objects) and System.threading (for multi-threaded applications) and System.Windows.Forms (which allows for quick development of UI).

2.2.4.3 Comparison with J2EE

.NET main competitor in the application server market is the java offering of J2EE. Market share is increasing for .NET over J2EE for a number of reasons even though .NET will not come for free.

2.2.4.3.1 Interoperability

Both have web services within their scope. These are supported through XML and SOAP on both systems. Both Microsoft and Sun have made great efforts to create common web standards.

2.2.4.3.2 Security

Any windows environment brings with it some security flaws. This is not so much down to the software but more down to the level of effort put in by the hacker. The .NET framework has security features built into its core but here is where J2EE has a stronger position.

2.2.4.3.3 Maintainability

Coding using J2EE is a much more difficult job than using the .NET framework. With its complex JavaBean system which increase security and separation of concerns it also makes maintaining or upkeep of the code something only someone with a high level of java development skills can do. .NET's high integration with the operating system makes a lot of services available to the programmer. This makes it easier to maintain along with a more simple coding style.

2.2.4.3.4 Scalability

J2EE wins here because of the fact it is operating system independent. This allows for greater growth as the .NET framework is tied to the Windows environment.

2.2.4.3.5 Cost

J2EE comes off certain vendors for free. .NET has a licensing system which will cost the end user. The end user will also have to run a Windows environment for their .NET application.

2.2.4.3.6 Language Support

.NET wins here with a vast array of languages supported through the .NET framework. Only java is supported natively by the J2EE.

3. Design and Implementation

3.1 Lifecycle and Process

The development process undertaken during the construction of this project was based on the "Waterfall" process. The waterfall process is what is known as a sequential software development process. It's first mention was by "Winston W. Royce" in 1970. (IBM)

The 5 core stages of the waterfall process are as follows.

- Requirements
- Design
- Implementation
- Verification
- Installation and Maintainence

For the waterfall process to be a success it is neccessary that all stages previous were implented perfectly. There is no room for error. This is unfortunate as errors are something that us humans do very well!

Arguements for the waterfall process claim that more time spent on the earlier stages of the cycle will lead to greater economy in the later stages. While this is true - and something that is true of all development cycles it can lead to over emphasis on requirements and design. This in turn can lead to little time or money to continue with the implementation and verification stages in a real world environment.

It has been proven that while this process is useful for other engineering sectors in the realm of software engineering it is considered bad practise. Software engineering demands a more iterative approach to iron out bugs, involve the end user in participatory studies, conforming to standards and redesign.

With this in mind it was impossible for me to consider any other process due to time constraints. Exploratory research into the structure of PDF, comprehension of the wealth of functionality of the .NET framework lessened the amount of time available for actual implementation.

3.2 Requirements

Gathering requirements for the project took much longer than expected. To build a list of requirements I started with an initial list built from requirements i wished my project to have as functionality. The main priorty i had was to create a truely touchscreen application. The other was to try and encorporate PDF functionality. I knew that this alone would not be the sole requirements list . To gather more I built use cases from the list of actors the system would have to have.

3.2.1 Actors

An actor is "an external entity of any form that interacts with the system. Actors may be physical devices, humans or information systems" (Bennett, McRobb, & Farmer, 2006)

The following is a list of the actors involved in the application:

Actor	Description
Student	The student can download the latest worksheet for the module. The student can answer the worksheet. The student can upload the answered worksheet.
Teacher	The teacher can download worksheets from the server that students have uploaded. When the teacher has finished correcting the worksheet they can upload the corrected sheet back to the server and register the score achieved by the student.
Server	The server must be able to upload and download files to student. The server must be able to upload and download files to teacher. When uploading files from teacher the server must maintain record of student score received.

From this list of actors i was able to create a list of use cases for the application.

3.2.2 Use Cases

"A use case is a description of a set of sequences of actions, including variants, that a system performs to yield an observable result" (Booch, Rumbaugh, & Jacobson, 1999)

"A use case describes, from a user's perspective, a behaviorally related set of transactions that are normally performed together to produce some value for the user" (Bennett, McRobb, & Farmer, 2006)

Essentially, a use case describes an action by an actor and what happens with the system or application when that action occurs.

Using this process i was able to gather a bigger list of functional requirements of the application. It is a very good way of discovering functionality which may not be clear at concept of idea.

Listed in the appendix is some early use cases, descriptions and requirements.

3.2.3 Non Functional Requirements

Often forgotten in the development of applications is the consideration of non functional requirements. Non functional requirements cover a depth of areas which do not directly affect the funtionality of a product. Here are the two key non functional requirements for this application.

3.2.3.1 Useability

I was aware of some of these before i began the project. One of my major concerns was the "Human Computer Interaction" factor. Seeing as it is a touch screen application interaction with the interface is different than your normal application. I had to minimise the use of keystrokes or input from a keyboard. While the Windows operating system caters better than other for the use of tablet pc's most applications fail to provide suitable interfaces. After consulting the UX guide (guide) i came up with the follow features.

Some key features for touchscreen applications are as follows:

- Minimal input to be required from tradition input devices such as keyboard.
- Large easily identifiable buttons.
- Use of information bubbles to describe features of the application. This facilitates minimal use of menus.
- To try and provide useable shortcuts on the main interface. This again minimises use of menus.
- That the application is adaptable to change in screen resolution. Most touch screen computers have the facility to turn into tablet mode where portrait mode is a more manageable format (Figure 6).

To overcome some design flaws before implementation i used Microsoft Visual Blend to design and mock up some user interfaces.

3.2.3.2 Security

In the modern era as computers become more essential in an educational setting it is imperative that security is considered in any application regarding test scores or test papers. As has been proved countless times hacking is an issue which any website administrator or database administrator constantly has to guard against.

In my application every user has to autenticate through a login. In the database all current users of the system have a password which was tied to their unique identifer.

3.3 Architecture of System

The architecture of the application is an "n tiered system". Loosely based on the concept on the Model View Controller pattern the system encorporates a seperate user interface , an appllication server to facilitate the user interface and communicate with external systems, a web server and an enterprise information system. It differs from the Model View Controller pattern as it is not cyclical in nature, but rather linear. Each layer of the application talks to those immediatly around it and no further. While computational costs of such an architecture are expensive, in today's multi core, large memory capacity computers this cost is not a consideration that has to be placed as highly on the list of functional requirements.

The benefits of such a system are multiple. Firstly each system can be chosen for which best meets its requirements. This in turn also causes seperation of concerns which again is beneficial for the longevity and uselfulness of an application.

3.3.1Implementation Technologies Used in the Application

There are six main technologies being used in the development of this application.

- Microsoft Visual Studio 2010
- C# with WPF
- PHP
- MySQL
- Apache
- Ghostscript
- Microsoft Visual Blend

Microsoft Visual Studio 2010 was used because the application is written in the C# language. Improvements over Visual Studio 2008 are better capabilities in rendering the XAML language. XAML (**Error! Reference source not found.**) is a newer form of the informs namespace with greater scope for individualization of the user interface and buttons on it.

C# with WPF was used because the .NET languages work in a Window environment. Currently Microsoft operating systems far exceed other operating in dealing with touch screen technology. WPF (Windows Presentation Foundation) is a recent edition to the Microsoft IDE. It is used in this project because of the array of functionality it provides for tablet pc developers.

PHP (PHP) is an open source language which is a dominant player in web development. The scope of its use in this project is limited to dealing with the transfer of files to the server. It will also deal with verification of user

MySql (MySql) is an open source database recently bought by Sun Microsystems. It is used to store the files created by the application.

Apache (AP) is an open source web server . Known for its stability it is the market leader in web server technology. It has 58.3% market share currently (netcraft).

Ghostscript (GS) is an interpreter for the PostScript language and for PDF. It is an open source application available for use under the GNU licence. It is used to render the PDf files to facilitate inking capabilities of the application.

Microsoft Visual Blend was used for creating prototype interfaces. Using this gave me completed XAML code which could be encorporated directly into Microsoft Visual Studio.

3.3.2 API for .NET Architecture

C# itself cannot create or render PDF. For this an external API must be used. I researched two APIs in depth, PDFSharp and Itext.

3.3.2.1 PDFSharp and MigraDoc

PDFSharp (PDFSharp)is an open source .net library for creating and processing PDF files on the fly from various .net languages. MigraDoc renders documents to help create PDF documents. It includes support for C/C++, VB, C # and WPF. It was created by **empira Software** of Germany.

PDFSharp is entirely written in C#. The current version is 1.3. It was released in August 2009.

PDFSharp has a vast array of functionality for creating and modifying PDF documents. Main features include:

- C# object model to create PDF documents.
- Ability to split, merge and modify pages in PDF files.
- Can display PDF files through a viewer by invoking Adobe Reader.
- Can draw on existing PDF files.
- Can convert XPS files to PDF file.
- Can invoke Adobe Reader to display PDF files.
- Supports annotations on PDF files.

3.3.2.2 ITextSharp

ITextSharp (ITextSharp)is a port of the successful open source java library called IText. Originally developed by Bruno Lowagie it has developed over time to become one of the main open source libraries for creating and manipulating PDF documents. Similar to PDFSharp it can be used to create and modify PDF documents.

Main Features include:

- Serve PDF to a browser
- Generate dynamic documents from XML files or databases
- Use PDF's many interactive features
- Add bookmarks, page numbers, watermarks, etc.
- Split, concatenate, and manipulate PDF pages
- Automate filling out of PDF forms
- Add digital signatures to a PDF file

After exploratory research into both API's I decided to choose to use PDFSharp because of the greater documentation and community based support available. Another reason is has support for the WPF standard of C# development. While ITextSharp has support at its webpage and also has a book available it is strongly linked with the Java community and is also currently undergoing several changes. The project is being split into several different sections and is also changing its licensing system.

3.3.2.3 PDF Convert

This API gives the capability of being able to convert a PDF page to an image. I found this API on a popular C# programming website, www.codeproject.com.

A computer programming enthusiast going by the moniker by "Lord Tagoh" created this API in his spare time (CodeProject). With out the discovery of this API it would not have been possible to finish the creation of the application.

3.4 Analysis and Design

Designing the interface to the application was done in three stages. First a low fidelity version was drawn on paper. The size of the paper was equal to the size of the device on which the application would run.

Using Microsoft Blend a mid level (Figure 4) and high level (Figure 5) fidelity versions were created and tested on the application (msdn). Final interface used the portrait mode of the table to maximise writing area (Figure 6).

It was clear from the start that an object orientated design was fundamental to the completion of this project. When using C# it is easy practically impossible to stay away from an object orientated environment as each interface created has its own class.

Initial class construction was decided on by gradual implementation. Firstly the task of manipulating the PDF files was developed. This expanded to two classes as i divided the inital class so as to seperate manipulation of PDF files from the creation of new PDF files with ink capture.

Next was gaining the ability of storing the ink layed down by the stylus. This involved using the ink.strokes collection from the C# libraries (msdn ink). Manipulation of this collection is left to the programmer. I created a single class, inkhelper.cs, which dealt with updating the strokes collection.

Next was to implement a system where i could insert ink onto the PDF page, save it as a PDF and do it again to a saved document. After extensive research into the use of PDF layer technology time became a critical factor and i had to abandon the idea. The workaround solution i came up with was to use the libraries available to me within the .NET framework. Using Ghostscript and the PDFConvert API I captured a picture of the submitted PDF. This picture was then used in the imagebrush container provided by the .NET framework.

In the PDFPage class i combine the strokes collection and the image taken to create the new PDF with ink inserted on to it.

One of the biggest and most complex classes created was the webhelper class. This class had multlpe functions and functionality. It dealt with the creation of unique uniform resource identifiers so as to be able to pass requests to the php server.

It has functions which in combination with the php script on the web server verify who is logged in .

I implented a singleton design pattern in this class (msdn). Seeing as the whole application revolved around the information the instance of the class provided it was neccessary to repeatedly call it. Rather than have multiple instances of the same object the singleton pattern makes sure only one instance is in use. This enabled me to insure that all current data held in the webhelper class was the only one in existance within the application.

The webserver and enterprise information system were the less strenueous parts to set up and put into operation. The design of my tables in the database were poorly implemented which meant that providing a feature where modules and classes could be selected from were abandoned. It proved to me the value of well designed tables in a database.

3.5 Implentation Fragments

3.5.1 public void submit document()

First a query string is created with teh post data   "?upload=student"

This tells the php script that it is going to prepare itself for an upload from the student.

On 342 the uri for for the php upload is created.

From lines 334 to 340 create the headers for the HTTP Request to the server. The server must be told what type of file is coming . For this I created a multi part form data request to the server. This will tell the server to expect a file as well as data .

From line 351 to 364 is for the creation of a string which will become the header information to be appended to the request to the server.

On line 366 the string just created is appended onto the request to the server

On line 367 the header is encoded into UTF 8. This is because most servers expect headers in UTF8 format as oppose to ASCII.

On line 371 we create a file stream to upload the file. To do this we get the length of the file, the header and the boundary. This will inform the server of all the information needed to be able to receive the filestream. It will also be able to calculate expected upload time from the parameters given.

375 - create a new web request.

380 - Write the contents of the file to the stream. Once the headers are sent to the server the server will be waiting for the data of the request.

The data is then buffered into a byte array which is in turn written to the request stream. Once the stream is written to with the data a webresponce is opened with the server to get a responce from the server

```
       public void submitDocument()
        {
            string postdata = "?upload=student";
//334 to 340
            if (this.queryString != null)
            {
                foreach (string key in this.queryString.Keys)
                {
                    postdata += key + "=" + this.queryString.Get(key) + "&";
                }
            }
// line 342
            Uri uri = new Uri(this.server + "/inscribulus.php" + postdata);

            string boundary = "----------" + DateTime.Now.Ticks.ToString("x");
            HttpWebRequest webrequest = (HttpWebRequest)WebRequest.Create(uri);
            webrequest.CookieContainer = this.cookies;
            webrequest.ContentType = "multipart/form-data; boundary=" + boundary;
            webrequest.Method = "POST";
// line 351 to 364
            // Build up the post message header
            StringBuilder sb = new StringBuilder();
            sb.Append("--");
            sb.Append(boundary);
            sb.Append("\r\n");
            sb.Append("Content-Disposition: form-data; name=\"");
            sb.Append("userSubmittal");
            sb.Append("\"; filename=\"");
            sb.Append(Path.GetFileName(this.uploadFile));
            sb.Append("\"");
            sb.Append("\r\n");
            sb.Append("Content-Type: ");
            sb.Append("application/pdf");
            sb.Append("\r\n");
            sb.Append("\r\n");
// line 366 367
            string postHeader = sb.ToString();
            byte[] postHeaderBytes = Encoding.UTF8.GetBytes(postHeader);
            byte[] boundaryBytes = Encoding.ASCII.GetBytes("\r\n--" + boundary +
"\r\n");

// line 371
            FileStream fileStream = new FileStream(this.uploadFile, FileMode.Open,
FileAccess.Read);
            long length = postHeaderBytes.Length + fileStream.Length +
boundaryBytes.Length;
            webrequest.ContentLength = length;

            Stream requestStream = webrequest.GetRequestStream();

            // Write out our post header
            requestStream.Write(postHeaderBytes, 0, postHeaderBytes.Length);
// line 380
            // Write out the file contents
            byte[] buffer = new Byte[checked((uint)Math.Min(4096,
(int)fileStream.Length))];
            int bytesRead = 0;
            while ((bytesRead = fileStream.Read(buffer, 0, buffer.Length)) != 0)
            {
                requestStream.Write(buffer, 0, bytesRead);
            }

            // Write out the trailing boundary
            requestStream.Write(boundaryBytes, 0, boundaryBytes.Length);
            WebResponse responce = webrequest.GetResponse();
            Stream s = responce.GetResponseStream();
            StreamReader sr = new StreamReader(s);

            Console.WriteLine(sr.ReadToEnd());
        }

    }
```

3.5.2 class PDFPage

This is the class that is the core of application . While the PDFBook class does all the "heavy lifting" of creating , manunipulating and rendering of PDF files the PDFPage class is what brings the application together.

On instantiation of the class three private members are created. The System.Windows.Ink collection "StrokeCollection" is created. Another class call inkhelper.cs was created to manipulate the current state of the collection. The collection is similar to an array. It is an array for holding strokes received on the ink canvas.

The constructor of the class takes two arguments, pagePath and a boolean create.

PagePath is the location of the file that will be used in the second private member - ImageBrush.

The boolean is flagged if this is a new page creation. It will instantiate a new StrokeCollection object. Otherwise it will call the function loadStrokes(). This function checks to see if the collection exists. If it does not it create a new strokeCollection. If it does exist it loads the collection .

This means that the PDF page can be loaded again and again while the most recent collection. This enabled me the ability to create a save function in my application with updated PDF file .

```
class PDFPage
    {
        private StrokeCollection strokes;
        private ImageBrush page;
        private string pagePath;

        public PDFPage(string pagePath, Boolean create)
        {
            this.pagePath = pagePath;

            Uri uri = new Uri(pagePath);
            this.page = new ImageBrush();
            this.page.ImageSource = new BitmapImage(uri);

            if (create == true)
            {
                this.strokes = new StrokeCollection();
            }
            else
            {
                this.loadStrokes();
            }
        }
```

3.5.3 private void loadStrokes()

```
private void loadStrokes()
        {
            try
            {
                if (File.Exists(this.pagePath + ".ink"))
                {
                    FileStream reader = new FileStream(this.pagePath + ".ink", 
FileMode.Open);
                    this.strokes = new StrokeCollection(reader);
                }
                else
                {
                    this.strokes = new StrokeCollection();
                }
            }
            catch (Exception)
            {
                this.strokes = new StrokeCollection();
            }
        }
```

3.5.4 public string createPdf()

This function is part of the PDFbook class. As the name suggest this is where the creation of the PDF file takes place. The constructor takes in a parameter which corresponds to the current inkCanvas object being used.

This class calls the Migradoc library . it starts by creating an empty PDF document. The way i organised the system was for each page of the PDF a new Page object was created. In the PDFbook class an arraylist of PDFPage called pages.

To create the new PDF i cycled through the arraylist on line 103. As i loaded each page into the canvas i also loaded the stroke pertaining to that page. Then in line 110 to 112 i capture an image of the PDF page and the stroke together.

The image type used is .bmp

Once the image is captured i begin to create the new PDF document. I fill the empty container created earlier. The PDF is highly customizeable through the PDFSharp API. Line 127 to 133 all dimension of the PDF are set.

To insert the image into the PDF i use the PDFSharp class and create what is called a section. Then a paragraph is inserted into the section. Both are involved with the layout of the final PDF.

Finally the MigraDoc class is called to render document and create finished PDF.

Final cleanup is done and all temporary files are deleted.

```
public string createPdf(ref InkCanvas canvas)
        {
            // Create a new MigraDoc document
            string filename = "submitted.pdf";

            PdfDocument document = new PdfDocument();
            document.Info.Title = "PDFsharp XGraphic Sample"; // get title of document
            document.Info.Author = "Stefan Lange"; // get student id
            document.Info.Subject = "Created with code snippets that show the use of 
graphical functions"; // some title
            document.Info.Keywords = "PDFsharp, XGraphics"; // keywords

            int p = 1;

            Application.DoEvents();

            foreach(PDFPage page in this.pages)
            {
                canvas.Background = page.getPage();
                canvas.Strokes = page.getStrokes();

                Application.DoEvents();

                RenderTargetBitmap rtb = new 
RenderTargetBitmap((int)canvas.ActualWidth, (int)canvas.ActualHeight, 196d, 204d, 
PixelFormats.Default);
                rtb.Render(canvas);
                BmpBitmapEncoder encoder = new BmpBitmapEncoder();
                encoder.Frames.Add(BitmapFrame.Create(rtb));
                FileStream fs;
                if (!File.Exists("./temp" + p + ".bmp"))
                {
                    fs = File.Open(@"./temp" + p + ".bmp", FileMode.CreateNew);
                }
                else
                {
                    fs = File.Open(@"./temp" + p + ".bmp", FileMode.Truncate);
                }
                encoder.Save(fs);
                fs.Close();

                PdfPage pdfpage = document.AddPage();
                pdfpage.Height = 1754;
                pdfpage.Width = 1240;
                pdfpage.Size = PdfSharp.PageSize.A4;
                pdfpage.TrimMargins.All = PdfSharp.Drawing.XUnit.Parse("0");
                XGraphics gfx = XGraphics.FromPdfPage(pdfpage);
                gfx.MUH = PdfFontEncoding.Unicode;
                gfx.MFEH = PdfFontEmbedding.Default;
                Document doc = new Document();

                Section sec = doc.AddSection();
                sec.PageSetup.TopMargin = 0;
                sec.PageSetup.BottomMargin = 0;
                sec.PageSetup.LeftMargin = 0;
                sec.PageSetup.RightMargin = 0;

                Paragraph para = sec.AddParagraph();
                para.AddImage(@"./temp" + p + ".bmp");
                para.Format.Alignment = ParagraphAlignment.Left;
                para.Format.Borders.Distance = "5pt";

                MigraDoc.Rendering.DocumentRenderer docRenderer = new
DocumentRenderer(doc);
                docRenderer.PrepareDocument();
                docRenderer.RenderObject(gfx, XUnit.FromCentimeter(2),
XUnit.FromCentimeter(2), "12cm", para);
                p++;
            }

            while (p > 0)
            {
                File.Delete("./temp" + p + ".bmp");
                p--;
            }


            document.Save(filename);
            return filename;
        }
```

4. Testing and Evaluation

“Software Testing is the process of executing a program or a piece of software with the intention of finding errors” (Myers, 1979)

This section due to an unfortunate clash of times will be much shorter than hoped for. I had hoped for actual input from a teacher working in a secondary school. Ellen has the fortune to work in a modern school with many technical tools available to her. It is unfortunate that I did not get her input as she has experience with the use of a electric whiteboard with overhead projector. Her input would have been invaluable.

I can talk about the type of testing undertaken during the design and development stages which i carried out myself. Any good software is tested to the best of the capabilities of the producer. It is impossible to cover all possible outcomes but most flaws can be found with extensive testing.

Because of the small development team unit testing at both a function and class level was the main form of testing done during development.

As the different components became completed i moved toward integration testing. This testing was done extensively once the PDF creation part of the project was complete. Inital integration testing of the server and database consisted of calls made to the database from the php script. Once the PDF files were created i used extensive integration testing to test the process. This revealed numerous bugs in my code which did not become apparent until then. The creation of the header for the php script needed to be converted to UTF-8 so the server could understand it.

Once the application was completed i conducted a series of load and performance testing. I made the system upload a large 50 page PDF file. This revealed that while it did not crash that it took considerable time to render the PDF to produce the images needed to create the background for the ink canvas. This also revealed that server storage capacity would be severly affected by large scale use of the system. The system withstood the load testing but performed poorly in the performance testing.

Finally smoke testing was carried out at the end of build to make sure that all functionality of buttons, PDF creation, manipulation ,ink collection and server requests and demands all worked properly. The only major area of failure was the re - rendering of the students PDF. This was due to using the two libraries PDFSharp and MigraDoc on an item they created previously. The type of image format also greatly affected the end product of the teachers PDF. While both libraries prove stable and had no crashes during use actual implementation.

Again because of the unfortunate timing of my subject i was not able to carry out user testing. If i had been able to do this i would have completed the "V model" of testing.

5. Discussion and Conclusions

It is hard to dicuss how i feel about the project. I feel the idea is an excellent one, one of the more usable projects presented on demo day. However the size and scope of the task became beyond my time frame. With demo day approaching i had to rush through certain parts which were pivitol to the concept.

I would have produce a better end product if i had concentrated on less criteria. The manipulation of PDF files is complex yet attainable with more time. I became fixated on the idea of using PDF layers to store the ink inputted by the user. If i had focused on this and had success then i would have created something very useful. There is no free version of PDF handwriting annotation. If i had managed to integrate layer technology i could have used the other features of PDF's such at radio buttons and javascript to make a truely interactive worksheet. I am however very happy with the amount of knowledge i have attained in the working of PDF files. It is a valuable attribute to have in the workplace.

A plus point of the experience for me was the exposure to the c# language. Once a person grasps the simple concepts of interface design supplied by the IDE programming happens quickly. The IDE is also a joy to work with. Academia's continued fight against Microsoft and it's technologies is questionable when the vast majority of young students have little exposure to any other operating system. There is a joy to creating an application with the c# language. It gives the user the opporunity to make a graphical interface with the simple drag of the mouse. In today's media rich society it makes the task of creating programs a little less mundane than the terminal echoing "hello world".

While the scope of the project was large i am glad to have had to deal with C# and the transferring of files to a server. Making this work was one of the most satisfying experiences of college.

I learned a great deal, and as i have discovered, in this industry you should never stop learning.

Bibliography

.net . (n.d.). Retrieved from http://msdn.microsoft.com/en-us/library/zw4w595w.aspx.

Adobe. (n.d.). Retrieved from http://www.adobe.com/uk/products/acrobatpro/?promoid=DTEVG

Adobe System Incorporated. (2006). *PDF Refernece.*

AP. (n.d.). *Apache*. Retrieved from http://www.apache.org/.

Apperley, B. P. (2007). *Making paperless work.* New York, USA: ACM .

bestpricecomputers. (n.d.). Retrieved from http://www.bestpricecomputers.co.uk/glossary/application-server.htm.

Bluebeam. (n.d.). Retrieved from http://www.bluebeam.com/web07/us/products/revu/standard/

CodeProject. (n.d.). Retrieved from http://www.codeproject.com/KB/cs/GhostScriptUseWithCSharp.aspx.

Evernote. (n.d.). Retrieved from http://www.evernote.com/about/home.php

Foxit . (n.d.). Retrieved from http://www.foxitsoftware.com/pdf/reader/

GS. (n.d.). *ghost*. Retrieved from http://pages.cs.wisc.edu/~ghost/.

guide, u. (n.d.). *http://msdn.microsoft.com/en-us/library/aa511258.aspx*. Retrieved from ux.

IBM. (n.d.). Retrieved from http://www.ibm.com/developerworks/rational/library/4626.html.

ITextSharp. (n.d.). Retrieved from http://itextsharp.sourceforge.net/

Jarnal. (n.d.). Retrieved from http://www.dklevine.com/general/software/tc1000/jarnal.htm

Jarnal. (n.d.). Retrieved from http://www.dklevine.com/general/software/tc1000/jarnal.htm

Microsoft OneNote. (n.d.). Retrieved from http://office.microsoft.com/en-gb/onenote/default.aspx

mobile. (n.d.). Retrieved from http://www.gottabemobile.com/.

Mock, K. *TEACHING WITH TABLET PC'S.* University of Alaska Anchorage.

msdn. (n.d.). Retrieved from http://msdn.microsoft.com/en-us/library/ee817670.aspx.

msdn. (n.d.). Retrieved from http://msdn.microsoft.com/en-us/library/ms998213.aspx.

msdn ink. (n.d.). Retrieved from http://msdn.microsoft.com/en-us/library/microsoft.ink.aspx.

Myers, G. J. (1979). *The art of software testing.* New York: Wiley.

MySql. (n.d.). *MySQL*. Retrieved from http://www.mysql.com/?bydis_dis_index=1.

netcraft. (n.d.). Retrieved from http://news.netcraft.com/archives/2010/01/.

Panagiotis Siozos, G. P. (2008). Computer based testing using ‘‘digital ink”: Participatory design of a Tablet PC based assessment application for secondary education. *www.elsevier.com/locate/compedu* .

PDFAnnotator. (n.d.). Retrieved from http://www.ograhl.com/en/pdfannotator/

PDFSharp. (n.d.). Retrieved from http://www.pdfsharp.net/

PHP. (n.d.). *PHP*. Retrieved from http://www.php.net.

PLATO. (n.d.). Retrieved from http://en.wikipedia.org/wiki/Touchscreen.

Price, B. a. (1997). *Teaching programming through paperless assignments: an empirical evaluation of instructor feedback.* New York, USA: ACM.

Sim, G. H. (2004). Implementation of computer assisted assessment: Lessons from the literature. *Association for Learning Technology Journal* .

wikipedia. (n.d.). Retrieved from http://en.wikipedia.org/wiki/Portable_Document_Format

Willis, C. L. *Tablet PC’s as Instructional Tools or the Pen is Mightier than the ’Board!* University of Houston.

Xiang, W. *USE OF WIRELESS TABLET PCS AS AN EFFECTIVE LEARNING AND TEACHING ENHANCEMENT TOOL.* Queensland: University of Southern Queensland, Toowoomba, QLD 4350, AUSTRALIA.

Appendix A: Use Cases

A	Teacher upload PDF to server	Teacher creates work sheet. When completed teacher uploads worksheet to server for student to receive.
B	Student downloads PDF to application	Student logs in with unique identifer and password. When new worksheet is uploaded to server student is notified by server. A message will appear in application to tell student of new worksheet. Student will download worksheet and application will open in editor.
C	Student uploads PDF to server	When student has finished worksheet student will upload worksheet to server for correction.
D	Teacher dowloads answered worksheet	When student has uploaded worksheet teacher will be notified by application that there is new worksheet to be corrected. Teacher downloads and corrects answer sheet.
E	Teacher uploads corrected answer sheet	Upon completing correcting of answer sheet teacher will upload corrected answer sheet and marks attained by student. Marks will be associated with the student in the database.
F	Server receives file from student	Server will verify which student is uploading file through authentication of user by ID and password. Upon verification server will upload file and store in database.Server will notify relevant teacher of module that student has completed worksheet.
G	Server receives new worksheet from teacher	Server verifys teacher with ID number and password. Server receives new worksheet and inputs into database. Server notifies relevant students of module that there is a new worksheet.
H	Server receives corrected worksheet from teacher	Server receives corrected worksheet from teacher. Server updates the student's scores in the database with new score. Server stores corrected worksheet in database for records.

Appendix B: Requirements

No		Use Case
1	Application must be truely touchscreen orientated. No input from any other method other than the stylus .	
2	Application will use PDF documents as the backbone of sytem.	
3	Application will allow for the input of digital stylus writing into PDF documents	
4	Application will handle transfer of files from student	B,C
5	Application will handle transfer of files from teacher	A,D,E,H
6	Application will store student score	H
7	Application will allow for student to save worksheet when not completed	
8	Application will notify student of new worksheet	G
9	Application will notify teacher of new uploaded answered worksheet	F

Appendix C: XAML

The following is a screen shot of a simple application whose interface is designed using XAML, followed by the XAML used to create it.

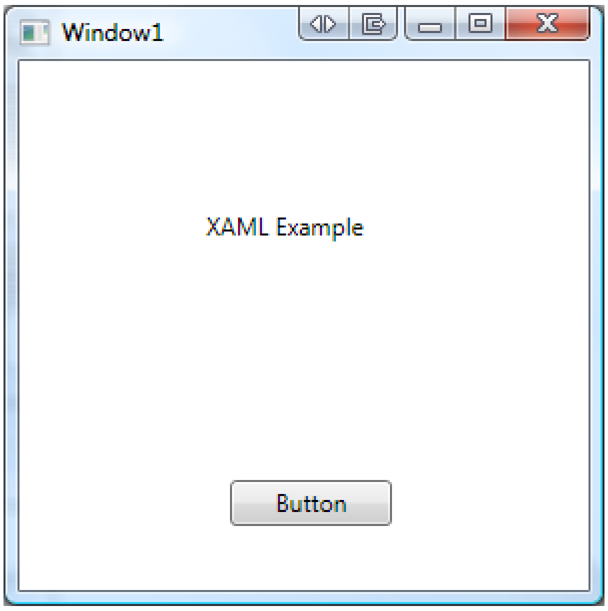


Figure 1

```
<Window x:Class="FYP_example.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Window1" Height="300" Width="300">
    <Grid>
        <Button Height="23" Margin="105,0,98,32" Name="button1"
VerticalAlignment="Bottom">Button</Button>
        <Label Height="28" Margin="88,69,70,0" Name="label1"
VerticalAlignment="Top">XAML Example</Label>
    </Grid>
</Window>
```

Appendix D: Images

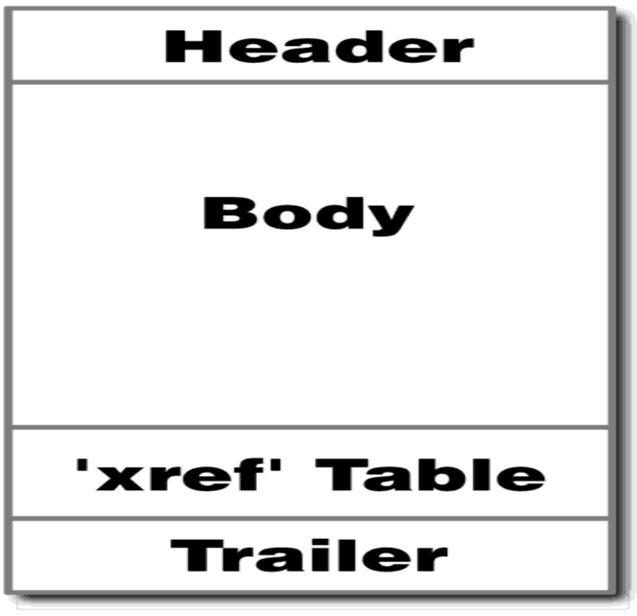
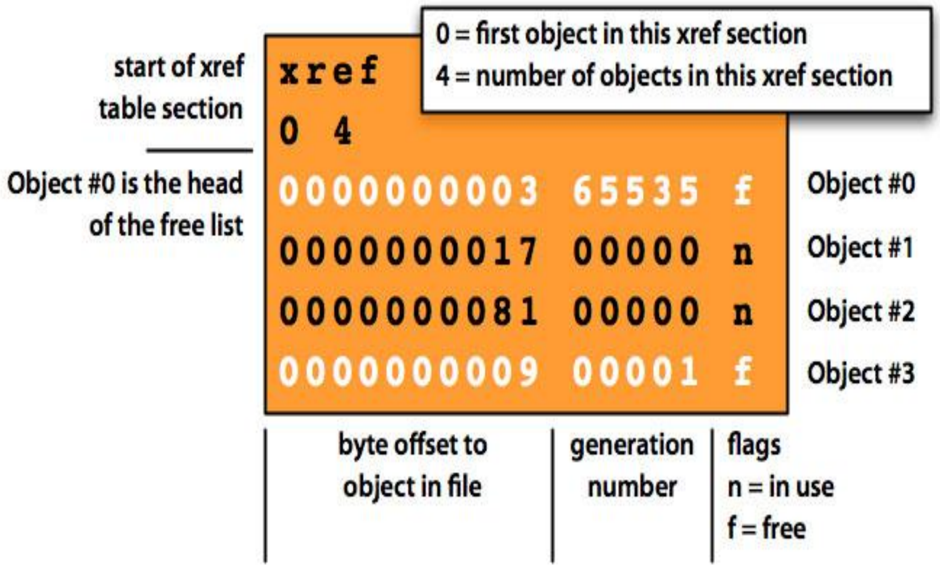


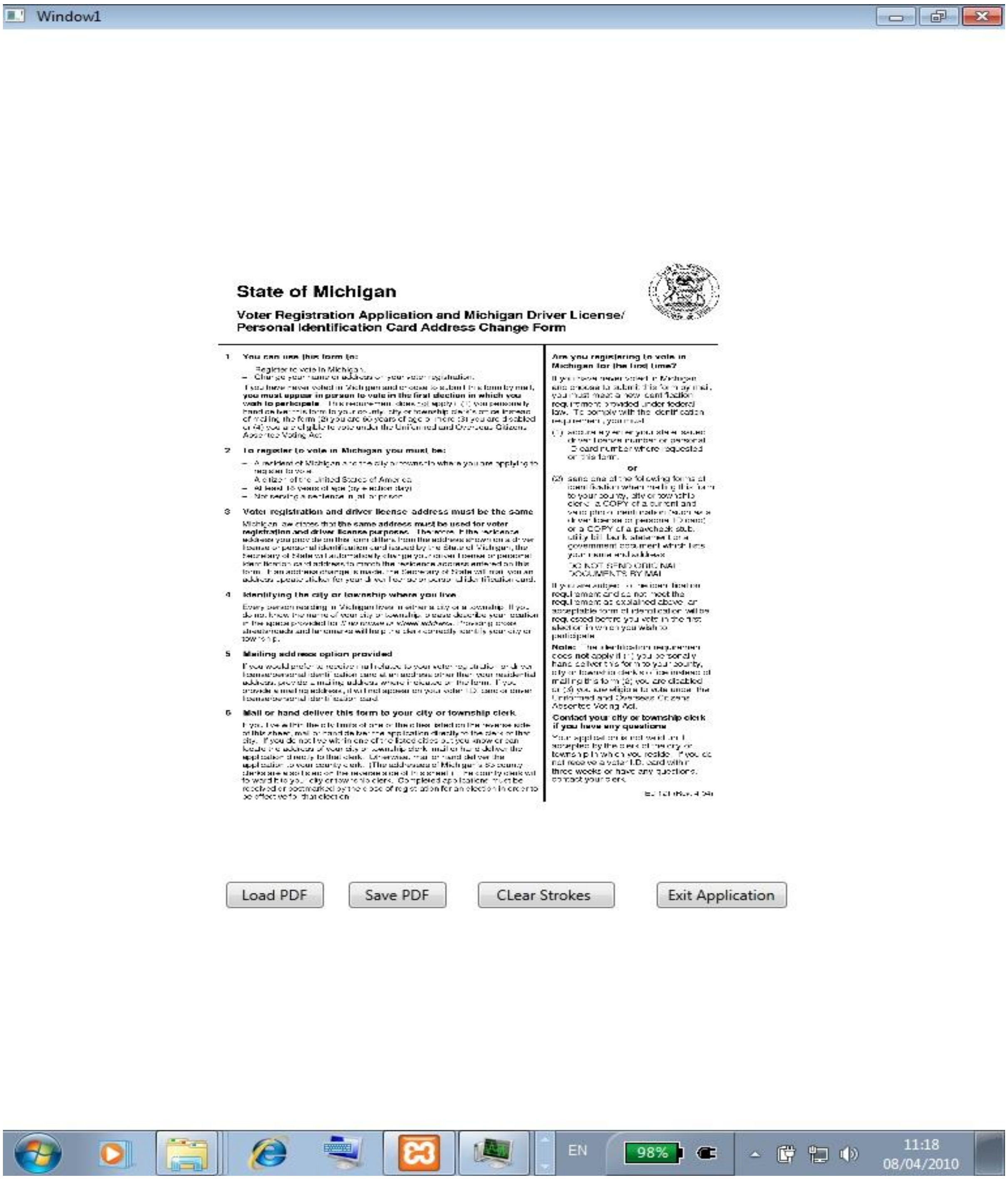
Figure 2


start of xref table section
Object #0 is the head of the free list
0 = first object in this xref section
4 = number of objects in this xref section
Object #0
Object #1
Object #2
Object #3
byte offset to object in file
generation number
flags
n = in use
f = free

```
xref
0 4
0000000003 65535 f
0000000017 00000 n
0000000081 00000 n
0000000009 00001 f
```

Figure 3 Xref table

Appendix E: Screen Shots


Window1

State of Michigan

Voter Registration Application and Michigan Driver License/ Personal Identification Card Address Change Form

1 **You can use this form to:**

- Register to vote in Michigan.
- Change your name or address on your voter registration.

If you have never voted in Michigan and choose to submit this form by mail, **you must appear in person to vote in the first election in which you wish to participate.** This requirement does not apply if (1) you personally hand deliver this form to your county, city or township clerk's office instead of mailing the form (2) you are 60 years of age or more (3) you are disabled or (4) you are eligible to vote under the Uniformed and Overseas Citizens Absentee Voting Act.

2 **To register to vote in Michigan you must be:**

- A resident of Michigan and the city or township where you are applying to register to vote.
- A citizen of the United States of America.
- At least 18 years of age (by election day).
- Not serving a sentence in jail or prison.

3 **Voter registration and driver license address must be the same**

Michigan law states that **the same address must be used for voter registration and driver license purposes.** Therefore, if the residence address you provide on this form differs from the address shown on a driver license or personal identification card issued by the State of Michigan, the Secretary of State will automatically change your driver license or personal identification card address to match the residence address entered on this form. If an address change is made, the Secretary of State will mail you an address update sticker for your driver license or personal identification card.

4 **Identifying the city or township where you live**

Every person residing in Michigan lives in either a city or a township. If you do not know the name of your city or township, please describe your location in the space provided for "Street number or street address." Providing cross streets/roads and landmarks will help the clerk correctly identify your city or township.

5 **Mailing address option provided**

If you would prefer to receive mail related to your voter registration or driver license/personal identification card at an address other than your residential address, provide a mailing address where indicated on the form. If you provide a mailing address, it will not appear on your voter I.D. card or driver license/personal identification card.

6 **Mail or hand deliver this form to your city or township clerk.**

If you live within the city limits of one of the cities listed on the reverse side of this sheet, mail or hand deliver the application directly to the clerk of that city. If you do not live within one of the listed cities but you know or can locate the address of your city or township clerk, mail or hand deliver the application directly to that clerk. Otherwise, mail or hand deliver the application to your county clerk. (The addresses of Michigan's 83 county clerks are also listed on the reverse side of this sheet.) The county clerk will forward it to your city or township clerk. Completed applications must be received or postmarked by the close of registration for an election in order to be effective for that election.

Are you registering to vote in Michigan for the first time?

If you have never voted in Michigan and choose to submit this form by mail, you must meet a new identification requirement provided under federal law. To comply with the identification requirement, you must:

(1) accurately enter your state issued driver license number or personal ID card number where requested on this form.

or

(2) send one of the following forms of identification when mailing this form to your county, city or township clerk: a COPY of a current and valid photo identification (such as a driver license or personal ID card) or a COPY of a paycheck stub, utility bill, bank statement or a government document which lists your name and address.

DO NOT SEND ORIGINAL DOCUMENTS BY MAIL

If you are subject to the identification requirement and do not meet the requirement as explained above, an acceptable form of identification will be requested before you vote in the first election in which you wish to participate.

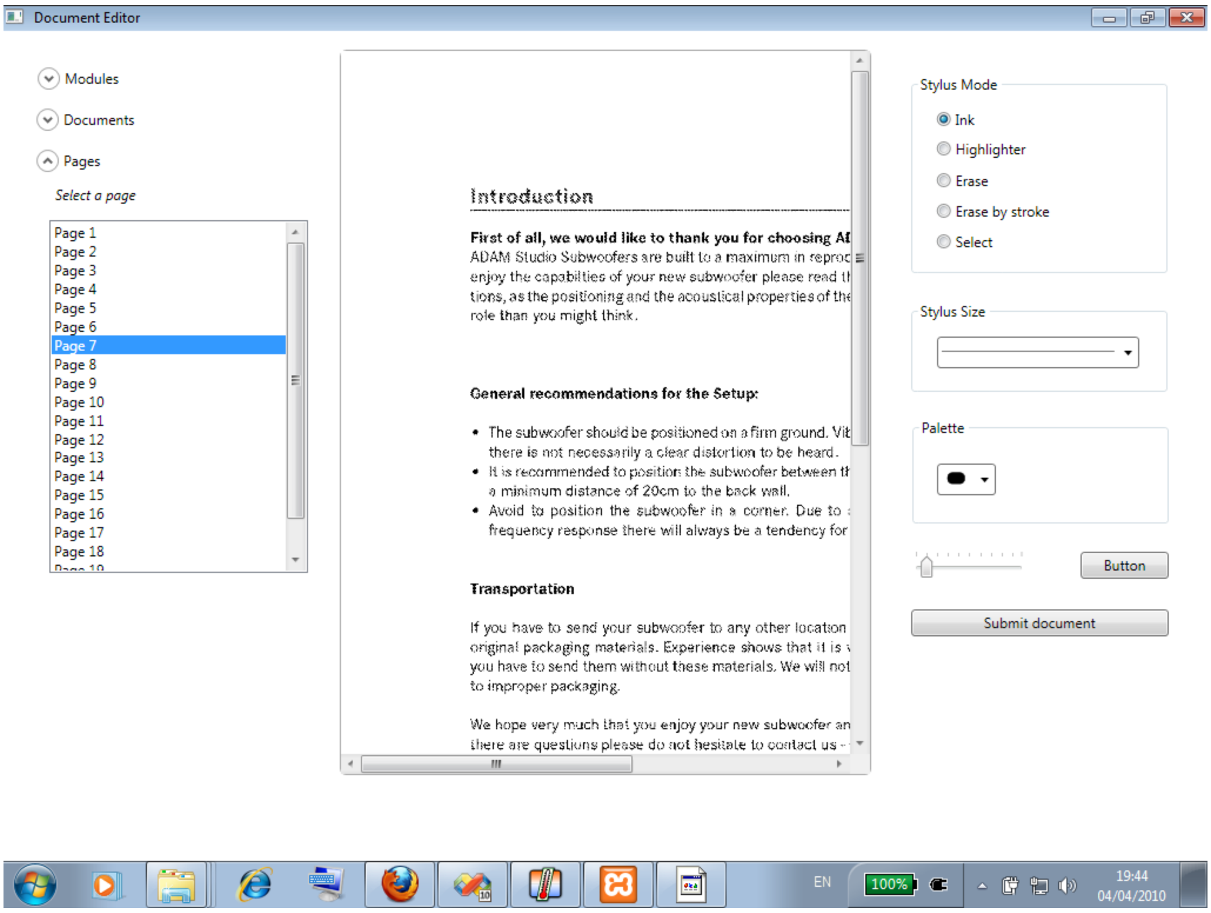
Note: The identification requirement does not apply if (1) you personally hand deliver this form to your county, city or township clerk's office instead of mailing this form (2) you are disabled or (3) you are eligible to vote under the Uniformed and Overseas Citizens Absentee Voting Act.

Contact your city or township clerk if you have any questions

Your application is not valid until accepted by the clerk of the city or township in which you reside. If you do not receive a voter I.D. card within three weeks or have any questions, contact your clerk.

ED 121 (Rev. 4/04)

Figure 4 First Prototype



Introduction

First of all, we would like to thank you for choosing AI
ADAM Studio Subwoofers are built to a maximum in reproc
enjoy the capabilties of your new subwoofer please read th
tions, as the positioning and the acoustical properties of the
role than you might think.

General recommendations for the Setup:

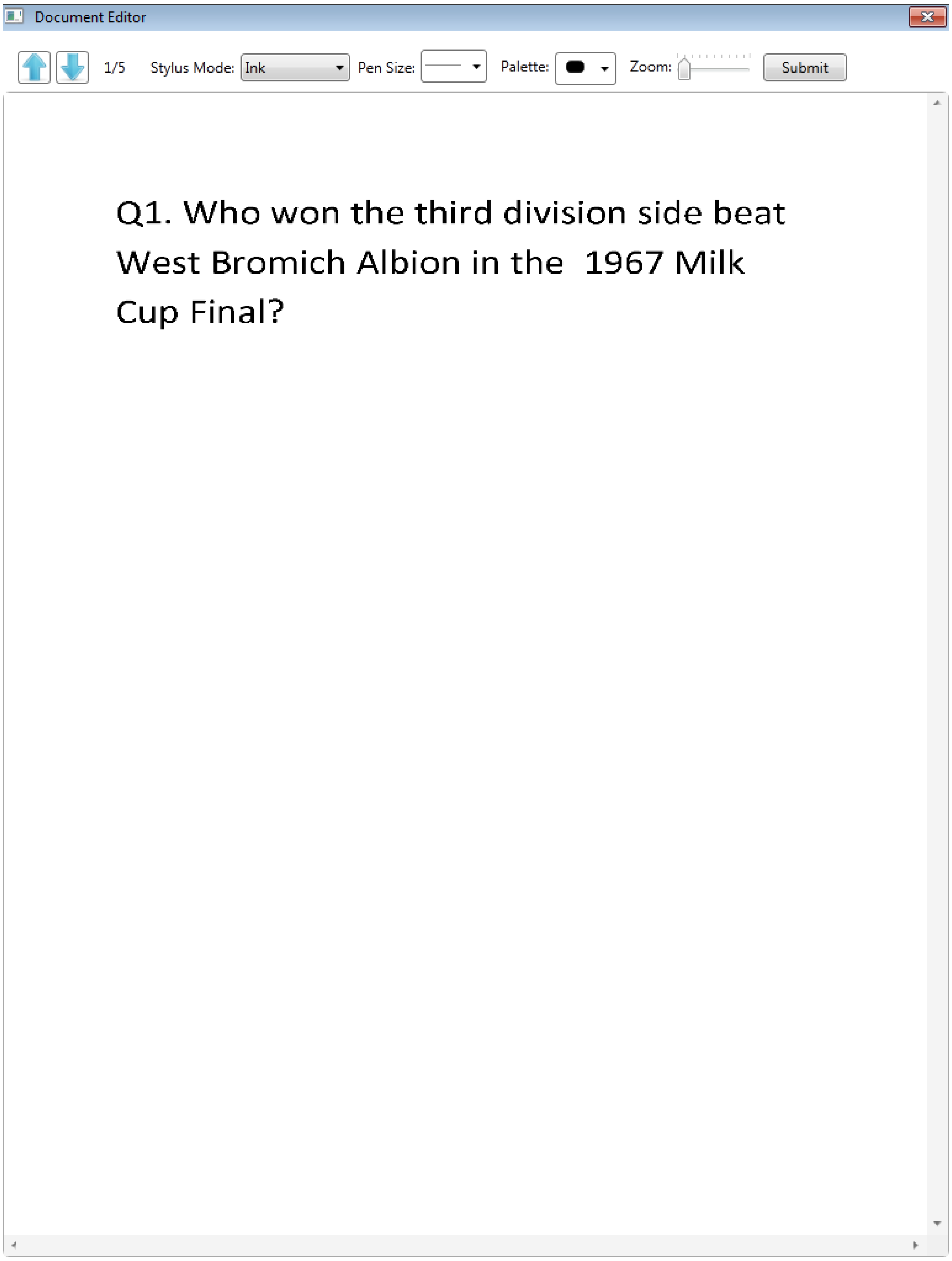
- The subwoofer should be positioned on a firm ground. Vit
  there is not necessarily a clear distortion to be heard.
- It is recommended to position the subwoofer between th
  a minimum distance of 20cm to the back wall.
- Avoid to position the subwoofer in a corner. Due to
  frequency response there will always be a tendency for

Transportation

If you have to send your subwoofer to any other location
original packaging materials. Experience shows that it is
you have to send them without these materials. We will not
to improper packaging.

We hope very much that you enjoy your new subwoofer an
there are questions please do not hesitate to contact us -

Figure 5 Second Prototype



Q1. Who won the third division side beat West Bromich Albion in the 1967 Milk Cup Final?

Figure 6 Final Student Interface

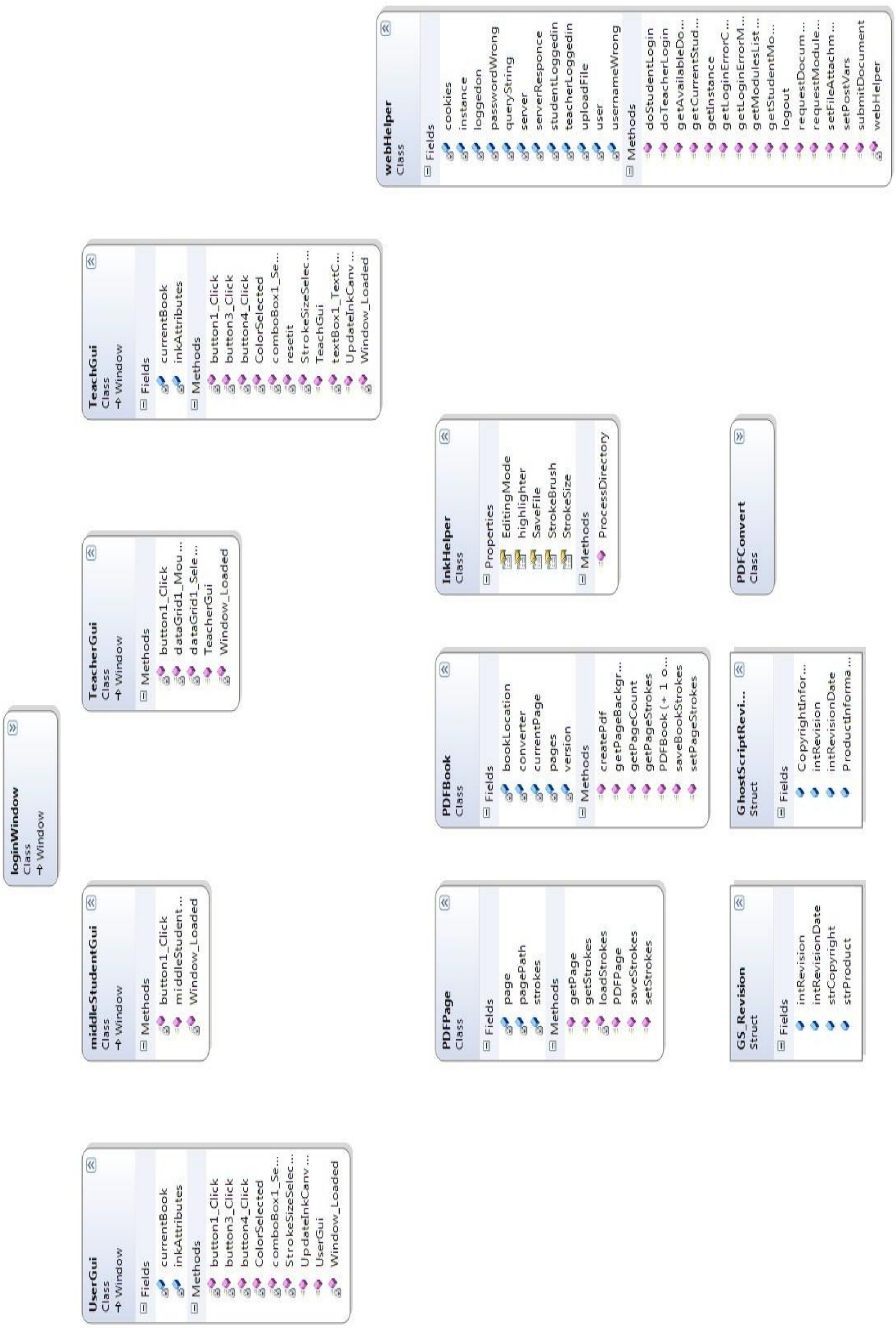


Figure 7 Class Diagram