FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

UNIVERSITY OF MALAYA

NAME : **NUR SURIA MANSOR**

MATRIC NO : **WET 020126**

THESIS : **INTERACTIVE SONG COMPOSER**

SUPERVISOR : **CIK MAS IDAYU MD SABRI**

PANEL I : **EN AMIRRUDIN KAMSIN**

PANEL II : **PN FAZIDAH OTHMAN**

TABLE OF CONTENTS

CHAPTER 1: PROJECT OVERVIEW

1.1 INTRODUCTION

The important of sounds as a channel of communication is emphasized and attention is drawn to some of its special properties. The success of the music industry and the importance of sound and music to film indicate how a substantial element of our culture relies on our sensitivity to sound, both to provide information and emotional effect.

Usually people develop musical skills through formal study and practical application. Much of the musical skills are absorbed unconsciously as part of the music-making experience. With the existent of recent multimedia technology, there are alternative ways to learn music in fun and interactive environment.

"Interactive Song Composer" is one of an interactive multimedia system that allows user to learn more about composing sound. It is stand alone system that allows user to select and combines the provided Midi's according to their creativity to produce a new kind of sound / MIDI.

1.2 PROBLEM DOMAIN

I have identified a few problems that have been stressed in the journals that I have studied. They are:

i. Every user has different kind of ability and creativity in composing sound.

ii. For music education, theory and lesson in classroom is not enough.

iii. Many of the interface requirements for a music system are unique to music and have no equivalent within the general realm of computer usage.

1.3 OBJECTIVES

Below are the objectives of this system:

i. Produce interactive multimedia system that can be used as learning tools for further music exploration.

ii. Produce song composer system that best suites for every kind of user level which is means from beginners to the higher level.

iii. It will allow users to compose sound according to their creativity levels.

1.4 SCOPE

1.4.1 User Scope

It is specially develop for totally beginners to learn basic level of compositing sound. There is no limit of age as long as they have interest in music or sound with basic computer knowledge.

1.4.2 Project Scope

This system is a stand alone that can be used without requiring any other additional support software. Basically the system is divided into two major parts:

- The design and architecture part
- The engine that will be running behind the system process – programming/coding

1.5 MOTIVATION

- Contributing the construction a tools for education or public purposes especially in music fields. This is for better understanding and simplifying the works related to music field as to experience by their own how it's like composing a song.
- To apply the skills in practical not only just knew theories. Not to waste the theories learned in computers field.

1.6 EXPECTATION OUTCOME

Outcomes:

- This system could be easily being understood by the user especially on how it works and how to use the overall application seems that this system is developed focused on the beginners.
- This software could be use as teaching-aid in classroom environment.
- User will find its overall interface and presentation is interactive and simple.

CHAPTER 2: LITERATURE REVIEW

2.1 LITERATURE REVIEW

Literature review is one of the most important processes in the development of a system or software. It is crucial in order to provide system developers with the understanding of the system as a whole before they can really start developing the system. It is also essential as a medium to convey the initial idea of the system that is developed to the users, so that they will have the right expectation on the system.

The approaches that have been used throughout the implementation of this literature review are as follow:

- Fact Finding
- Searching for facts through read books, journals and over the internet
- Observation – observe and analyze existing systems and compare each other
- Collaboration – collaborate with experts in related fields

2.2 ARTICLES AND JOURNALS REVIEWS

2.2.1 Problem Statement

I have been done some research on music composition and music learning environment nowadays. The articles/journals that I have referred for my research are:

i. Supporting Musical Composition by Externalizing the Composer's Mental Space

by Shigeki Amitani and Koichi Hori

In this journal, they have done a research on the analysis of cognitive processes in musical composition: what the cognitive processes in musical composition are like and how the process is affected when a representation of information is changed.

ii. Sound Compositions for Expanding Musicianship Education

by Thomas A. Regelski

In this journal they were more focusing on education issues regarding to sound compositions. They done some analysis of the history and failure of compositional pedagogy for developing musicianship, a new rationale for such pedagogy presented.

iii. Computer-Music Interfaces: A Survey

by Bruce W. Pennycook

This paper studied on the unique problems posed by the use of computers by composers and performers of music. It explains about the complexity of music conversion, the interaction of musical interaction with computer devises and discussed a bit about musical tasks.

From there, I have identified a few problems that have been stressed in the journals above. They are:

iv. Every user has different kind of ability and creativity in composing sound.

v. For music education, theory and lesson in classroom is not enough.

vi. Many of the interface requirements for a music system are unique to music and have no equivalent within the general realm of computer usage.

2.3 DISCUSSION OF TOOLS

2.3.1 Microsoft Visual Basic

The Microsoft Visual Basic is a programming system that is being used worldwide to create computer software components and applications. Visual Basic is an extremely versatile, powerful and complex development system.

Visual Basic is a Window application that can be used to develop special purpose applications for Windows operating system. The powerful feature of Visual Basic is its ease of use and simple interface. User can easily create an application by following three simple steps; creating the interface, setting the properties of each object on the interface and simple coding activities.

Visual Basic is especially useful for prototyping and rapid application development.

2.3.2 Reviews on Operating Software

2.3.2.1 Windows 2003

Windows 2003 was built on NT technology with the familiar interface of Windows XP. Windows 2003 features integrated web capabilities making it easier for user to connect to the internet anytime and anywhere.

User also has faster access to information and able to complete specific tasks more quickly and easily. It provides user with an easier way to work with files, work on the web or even work remotely.

Windows 2003 also helps protect user data that transmitted over a network. It uses IPsec to provide security for virtual private networks, which allow organizations to transmit data securely over the internet.

2.3.2.2 Windows XP

Windows XP Professional is built on the code base of Windows 2000. It features a 32-bit computing architecture with a fully protected memory model. Thus, it makes Windows XP Professional as the most reliable version yet.

Windows XP also introduces several new features, which one of them is the System Restore. The System Restore enables users and administrators to restore their computer to a previous state without the risk of losing data. It also provides internet connection firewall which can protect small businesses from common internet attacks.

2.4 EXISTING SYSTEM

When it comes to the comparison with the existing system nowadays, as we can see more of them are quite complicated to understand and to use by those that we can categorized as new in this field which is composing a song. Most of the software in the market is only suites for those that already have a knowledge or experience in using a similar software or system and not appropriate for beginners. As I state in early, I have done some observation and analyzed existing systems and compared each other and came to this conclusion.

2.5 SUMMARY

The literature review which had been carried out gave better understanding on why we need to develop this system and the important of it. It also reviewed some of the existing systems to identify the opportunities of improvements that can be exploited in the development of this song composer system.

By doing literature review, it can brings more knowledge general or specific to us. Where by doing this, study and research will be conducted to gather all the information. When number of research has been done, we didn't realize that we will automatically understand all the information which we never know before. And it is good for additional knowledge even we are not from the specific field.

CHAPTER 3: METHODOLOGY

3.1 INTRODUCTION

System development methodology is a very formal and precise system development process that defines a set of activities, methods, best practices, deliverables and automated tools for system developers and project managers to use to develop and maintain most of all software.

It serves as a guide throughout the system development process. It ensures that a consistent and reproducible approach is used from the first activity of the software development process until the software is completed. It also helps in reducing the risk of omitting important activities and consistent documentation is produced.

In this chapter, I will be reviewing on the system development methodology that had been chose for the development of this 'Interactive Song Composer' system.

3.2 RAPID APPLICATION DEVELOPMENT (RAD)

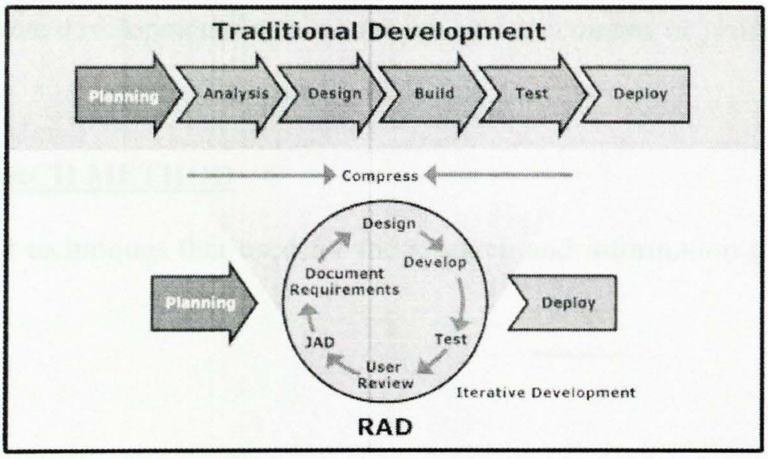


Figure 3.1 Rapid Application Developments (RAD)

Evolutionary development is based on the idea of developing an initial implementation, exposing it to user comment and refining it through many versions until adequate system has been developed. It enables developer to start with only part of the understood user requirements. The initial implementation will then evolve with the addition of features based on user feedbacks. The specification, development and validation tasks are conduct concurrently.

The advantage of this approach is that the specification for the system can be developed incrementally. It also provide user with better understanding of their problems. However, there is a risk that system will be poorly structured.

3.2.1 Why use RAD?

- To prevent cost overruns
- To prevent runaway schedules

- To limit a project's exposure to the forces of change
- To save development time - *at the expense of economy or product quality*

3.3 RESEARCH METHOD

The method and techniques that used for the research and information gathering or this project are:

- **Monograph**

Types of monograph that have been referred are books, journal and conference paper which is related to project domain. The journals and conference papers are mainly from university online library.

- **Internet sources**

Internet has been the main source of research especially in gathering the information regarding related system, additional information and in search of e-journals.

- **Previous thesis**

A lot of previous theses have been referred. By referring to these, it helps in giving a rough idea on the requirements of the final project paper and the issues that are needed in developing a system. It also helps a lot for better understanding the concept of this project.

- **Collaborations**

For better understandings and guidelines, collaboration has been made with my supervisor, Miss Mas Idayu Md Sabri whom giving me plenty of ideas on how to develop this system. She has consulted me on my project scope and her point of view has broadened my perspective on music in education and related issues. Also not to forgot, my project's partner, Miss Siti Rashidah Mat Som whom giving me a lot of help in sort of ways especially during brainstorming, to ensures this system is well developed at the end.

CHAPTER 4: SYSTEM ANALYSIS

4.1 INTRODUCTION

The next step after the literature search and review is to perform a detailed analysis. The main purpose of the system analysis phase is to learn exactly what takes place in the current system, and to determine and fully document in detail what should be included.

Through system analysis, the programmer or somehow called system engineer may manipulate system components toward the goal of improving the overall system. The information gathered during this phase has provided alternative strategies to develop this system. Through this phase also, the programmer can determine types of functional requirements and non-functional requirements for the system.

Following are some of the objectives of the analysis:

- To study the problem faced by the user.
- To study the problem and find out the best solution to reduced it.
- To acquire knowledge on how this system will be developed with the new emerging technology.
- To choose the development tools for the new system.
- To identify the major modules to be included in the system.

4.2 SYSTEM REQUIREMENT ANALYSIS

A requirement can be categorized as functional requirements and non-functional requirements.

4.2.1 Functional Requirements

Functional requirements describe the system's services and functions that provide for the users. This is based on the research and study that have been done.

- **Compose frame**
  - This section where the user begin to compose MIDI by mixed-and-matched the provided MIDI in the list.

- **Home frame**
  - This section is just for the introduction of the system and what it is all about generally.

- **User Manual frame**
  - Here, we provide a user manual in helping user to understand how to use or run the application appropriately step by step. all about generally.

❖ **Credits frame**

- Here, we state an acknowledgement to those who have been involved and participated in order to complete the development of this system.

4.2.2 Non-functional Requirements

Non-functional requirements are those constraints placed on the services or functions offered by the system (for instance, the required response time), or on the development process (such as the use of a specific language standard). These are important to ensure the quality of the system.

- **Flexibility**

  The system is able to adapt and implement in the changing environment in future.

- **Reliability**

  The system is reliable in perform its intended functions and operations accurately. For example, whenever any of the button is clicked, the system should able to respond and execute particular function accordingly such as generate some messages to validate user's action once again.

- **Usability**

  The system has been developed as simple as possible for user to use. Besides, the interface supposedly self-explanatory and consistent with other application in the system environment.

- **User friendliness**

  A good flow of navigation is important to help and guide users on navigating with little effort through hyperlinks and procedure steps. Good interface is able to improve interaction between the users and the system.

- **Efficiency**

  The system should be called or accessed in an unlimited number time to produce expected outcome or output at a creditable pace or speed.

- **Manageability**

  The system should be easy to manage and handle to ensure that maintenance can be done regularly. Besides, it should enable the evolutionary of the system easy to be done and making the enhancement works simpler.

- **Correctness**

  The system should be built according to the user requirements and specifications. It must meet its objectives and mission.

- **Maintainability**

  The system where the software should be able to be understood, corrected, adapted and allow enhanced in the future.

- **Expandability**

  The degree to where the architectural, data, or procedural design can be extended.

4.3 HARDWARE REQUIREMENT

Basically, the hardware requirements for 'Interactive Song Composer' are divided into two aspects, which is:

- System requirements for developer
- System requirements for user

Main hardware for developers (minimum requirements):

- ❖ Windows NT/2000/XP
- ❖ Intel Pentium 4 and above (or equivalent)
- ❖ Memory – 128Mb RAM or above
- ❖ 3.0GB hard disk or above

User's minimum hardware requirements:

- ❖ Intel Pentium 4 and above (or equivalent)
- ❖ Memory – 64MB RAM or above
- ❖ 3.0GB hard disk or above

4.4 SOFTWARE REQUIREMENT

Main software that has been use during the development of the system:

- Microsoft Visual Basic 6.0
- Microsoft Access
- Operating system: Windows XP
- Adobe Photoshop 7.0

CHAPTER 5: SYSTEM DESIGN

5.1 DATA FLOW DIAGRAM (DFD)

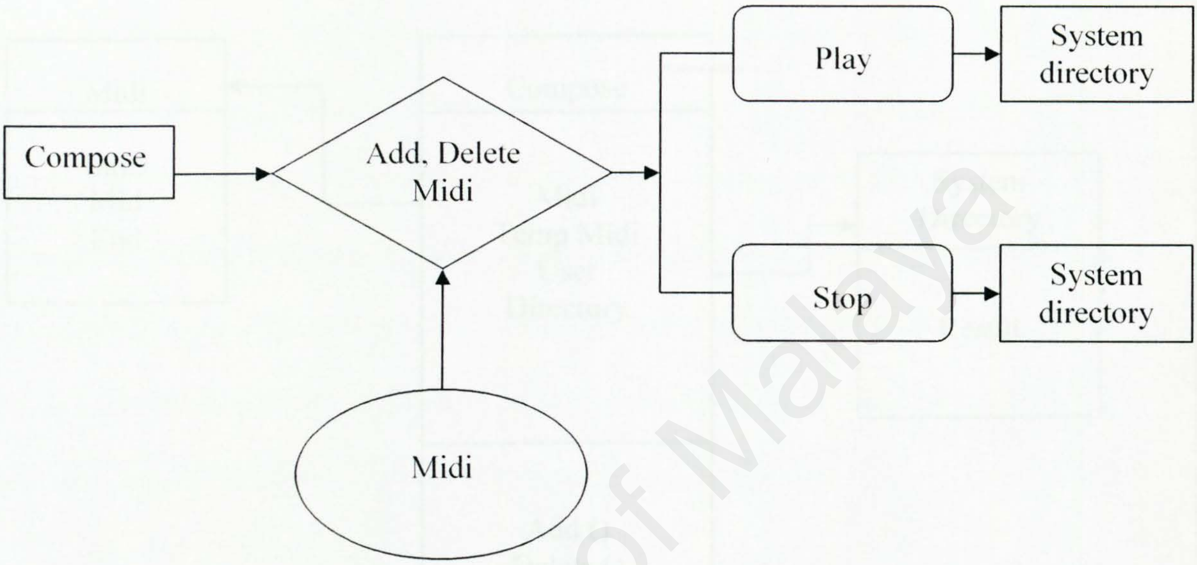


Figure 5.1: Data Flow Diagram (DFD)

Diagram above show the flow of data in the compose page. When user performing add () and delete () function on midi file (which are taken from database) user will choose what action to do next which are play or stop.

When user chooses to play the combining of those selected MIDI, the output will be the temporary file in database. Each time user performs this action the previous file will be deleted in the temporary file. The temporary file will only contain recent file produce by user.

If user want to save the output they made, a directory pop up will come out and user can choose which directory they want to save in the output.

5.2 OBJECT MODEL

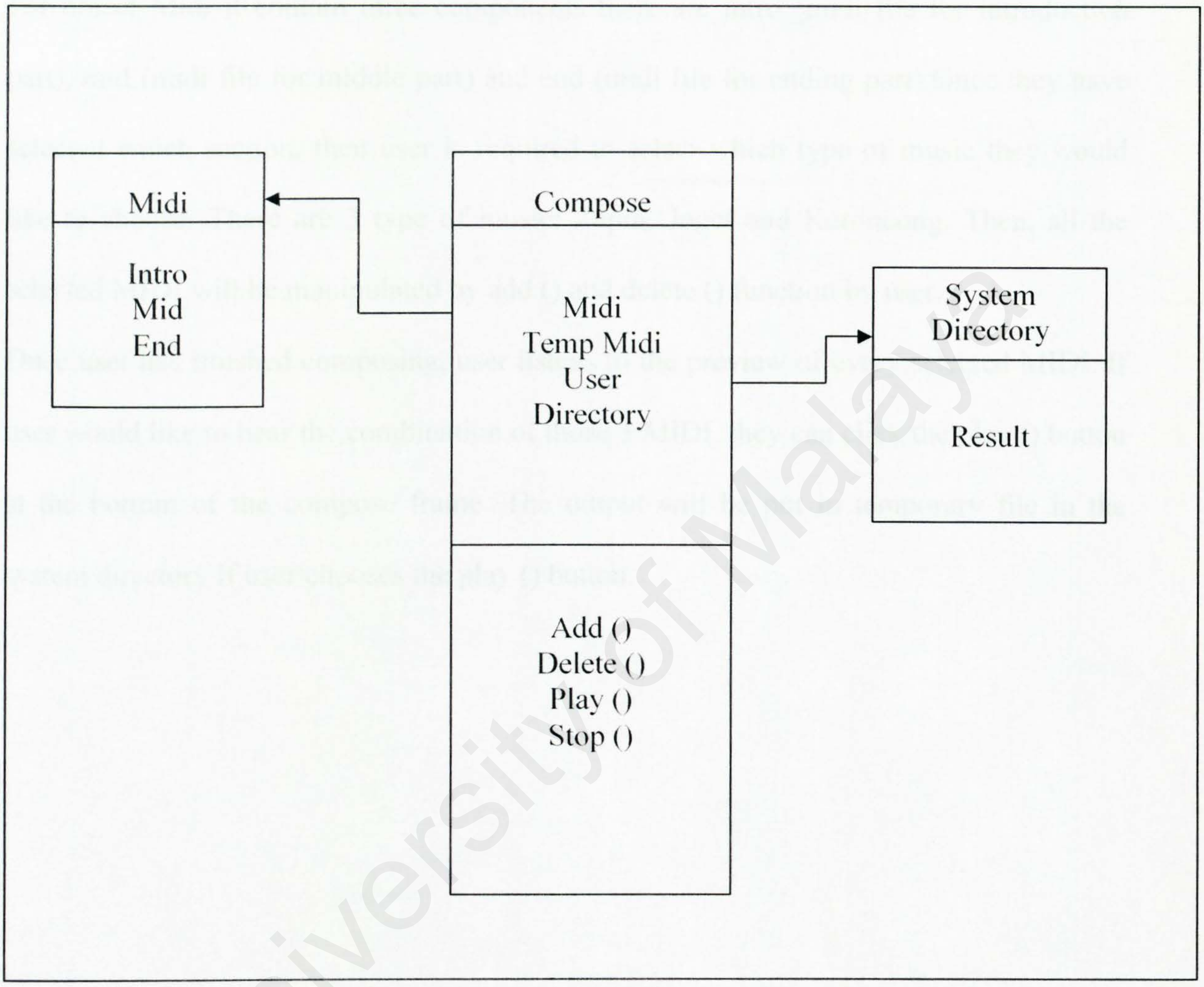


Figure 5.2: Object Model

Above diagram shows the relationship of object in compose page (page where user doing compositing activities).In this page there are five important function can be done, they are add (), delete (), play () and stop (). All the functions mentions are applied to the midi files that have been stored in database.

For object Midi it contain three components there are intro (midi file for introduction part), mid (midi file for middle part) and end (midi file for ending part).Once they have selected which section, then user is required to select which type of music they would like to choose. There are 3 type of music: Zapin, Joget and Keroncong. Then, all the selected MIDI will be manipulated by add () and delete () function by user.

Once user has finished composing, user listens to the preview of every selected MIDI. If user would like to hear the combination of those 3 MIDI, they can click the play () button at the bottom of the compose frame. The output will be put in temporary file in the system directory if user chooses the play () button.

5.3 SYSTEM MODEL

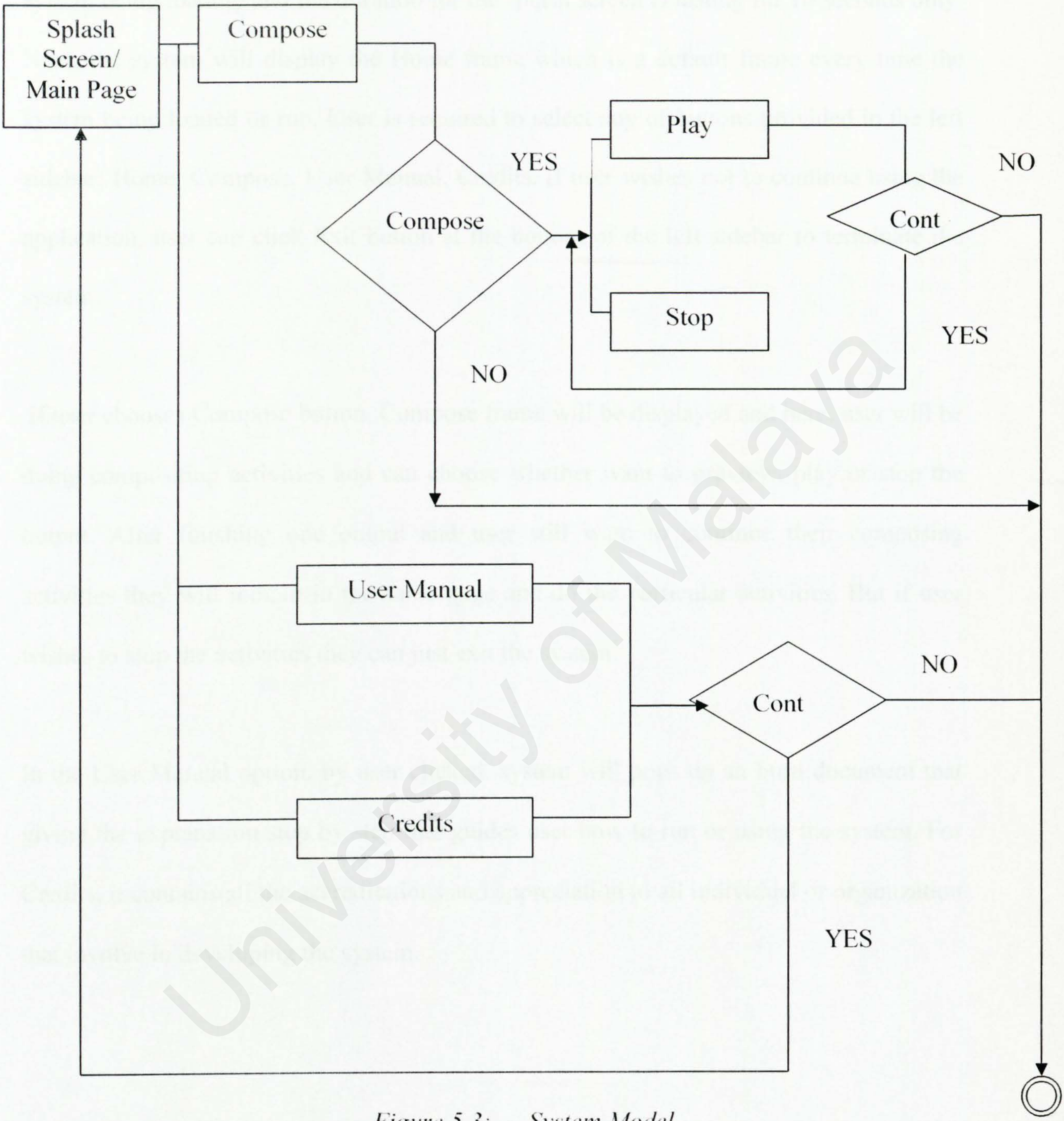


Figure 5.3: System Model

Figure 5.3 shows the model of the system. First, there will be a splash screen as the system being loading and the duration for the splash screen is lasting for 10 seconds only. Next, the system will display the Home frame which is a default frame every time the system being loaded or run. User is required to select any of buttons provided in the left sidebar: Home, Compose, User Manual, Credits. If user wishes not to continue using the application, user can click Exit button at the bottom of the left sidebar to terminate the system.

If user chooses Compose button, Compose frame will be displayed and here, user will be doing compositing activities and can choose whether want to preview, play or stop the output. After finishing one output and user still want to continue their composing activities they will remain in the same page and do the particular activities. But if user wishes to stop the activities they can just exit the system.

In the User Manual option, by user clicked, system will pops up an html document that giving the explanation step by step that guides user how to run or using the system. For Credits, it contains all the accreditations and appreciation to all individual or organization that involve in developing the system.

5.4 SYSTEM STRUCTURE

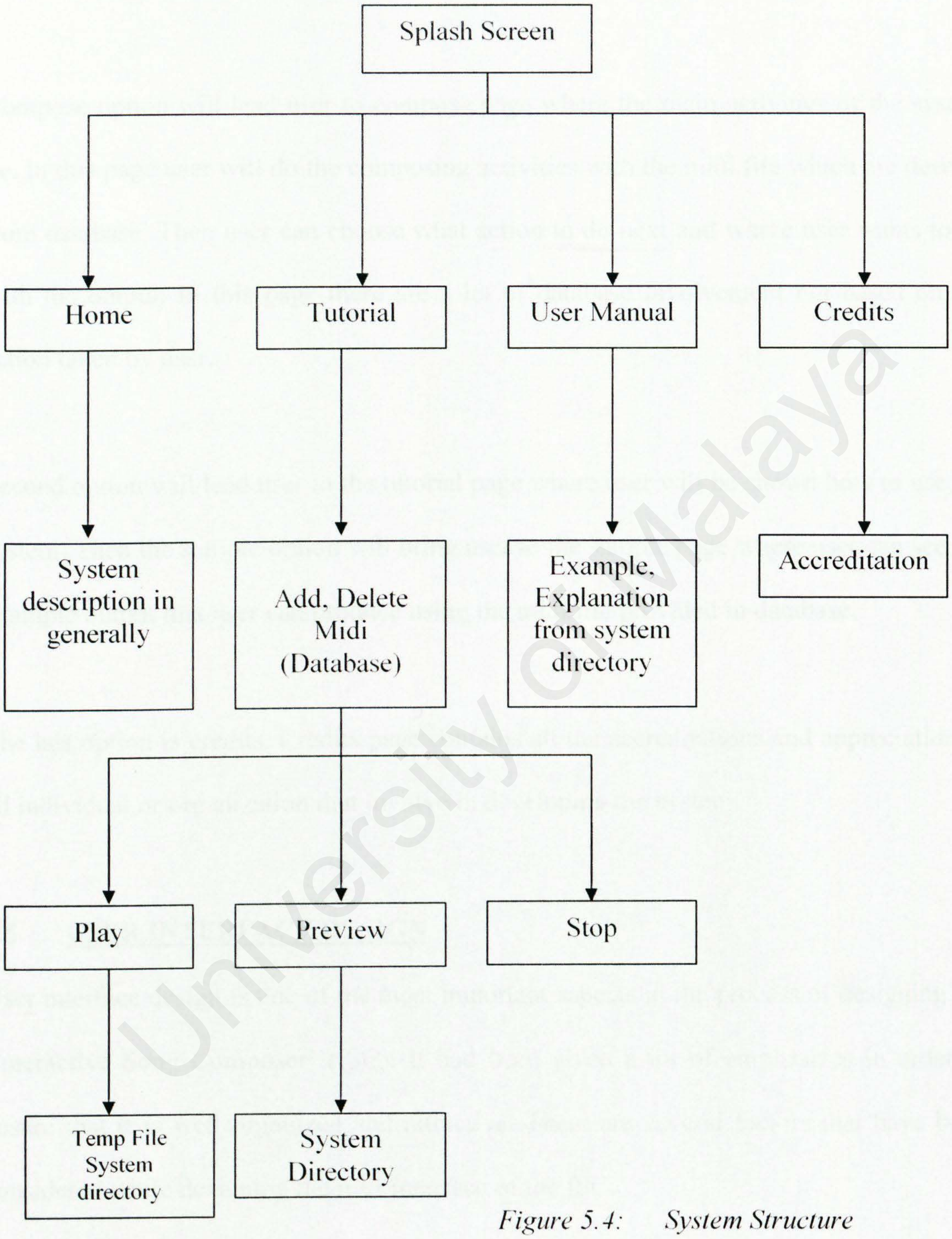

Splash Screen
Home
Tutorial
User Manual
Credits
System description in generally
Add, Delete Midi (Database)
Example, Explanation from system directory
Accreditation
Play
Preview
Stop
Temp File System directory
System Directory

Figure 5.4: System Structure

Figure 5.4 shows the structure of the system. The first page is index page which contain options of compose, tutorial, sample and credits.

Compose option will lead user to compose page where the main activities of the system lie. In this page user will do the composing activities with the midi file which are derived from database. Then user can choose what action to do next and where user wants to do with the output. In this page there are a lot of database involvement but based on the action taken by user.

Second option will lead user to the tutorial page where user will be shown how to use the system. Then the sample option will bring user to the sample page where user can see the example output that user can produce using the midi file provided in database.

The last option is credits. Credits page contains all the accreditations and appreciation to all individual or organization that involve in developing the system.

5.5 USER INTERFACE DESIGN

User interface design is one of the most important aspects in the process of designing the 'Interactive Song Composer' (ISC). It had been given a lot of emphasizes in order to ensure that it is well organized and attractive. There are several factors that have been considered while designing the user interface of the ISC.

1. Ease of use

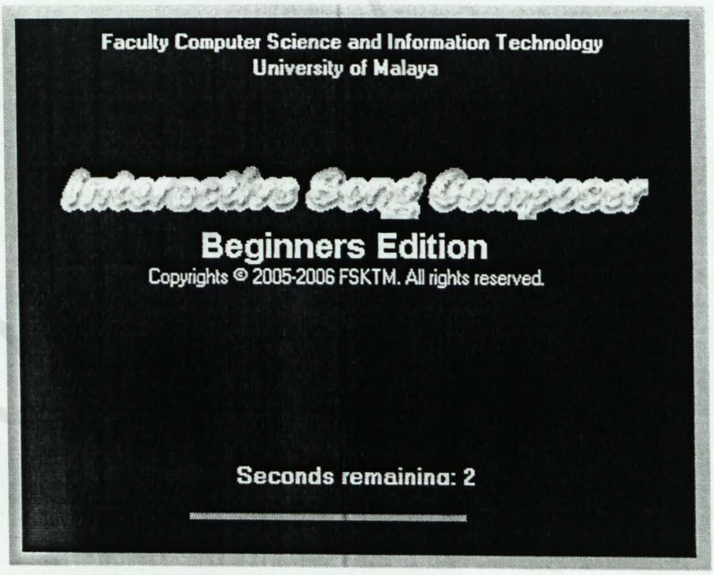
- The most important consideration in designing user is to ensure that the interface is easy to use
- As for the ISC, the layouts of the interfaces are designed to be consistent throughout the modules

2. Consistency

- An important element in the design the user interface. Through consistent interfaces, user will be able to use the system more effectively.

3. Performance issues

- To improve the performance, it is essential to ensure that user can easily understand and use the system. This could be achieved through a well organized and well guided interface.



Faculty Computer Science and Information Technology
University of Malaya

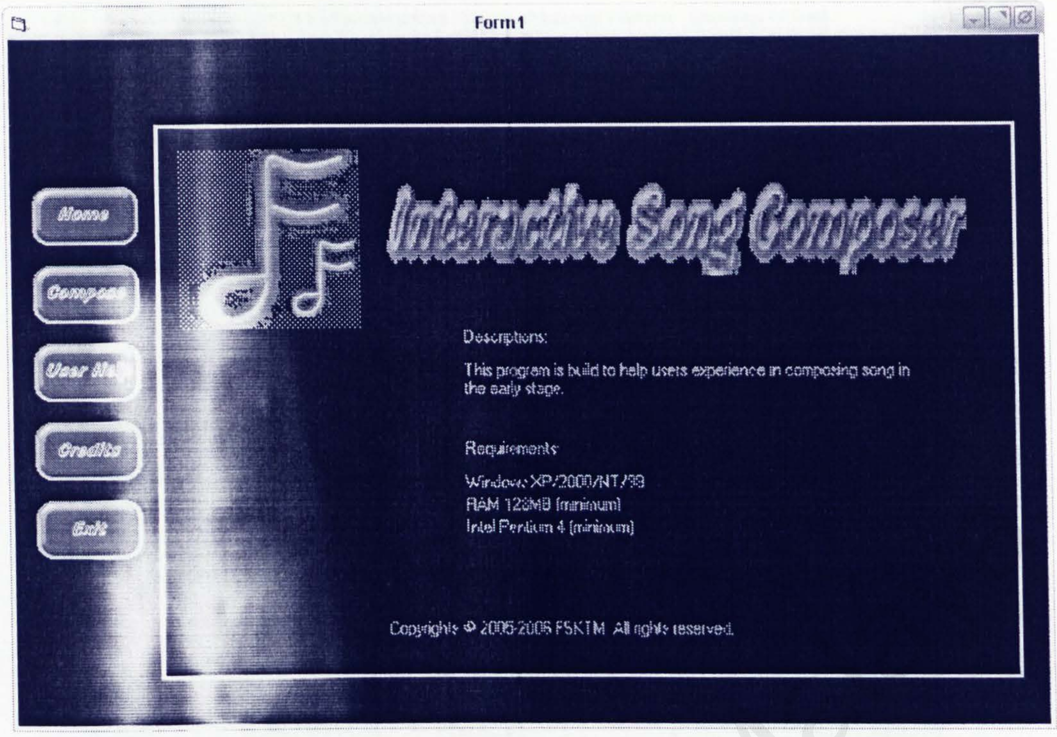
Interactive Song Composer

Beginners Edition

Copyrights © 2005-2006 FSKTM. All rights reserved.

Seconds remainina: 2

Figure 5.5 Splash Screen


Form1

Interactive Song Composer

Descriptions:

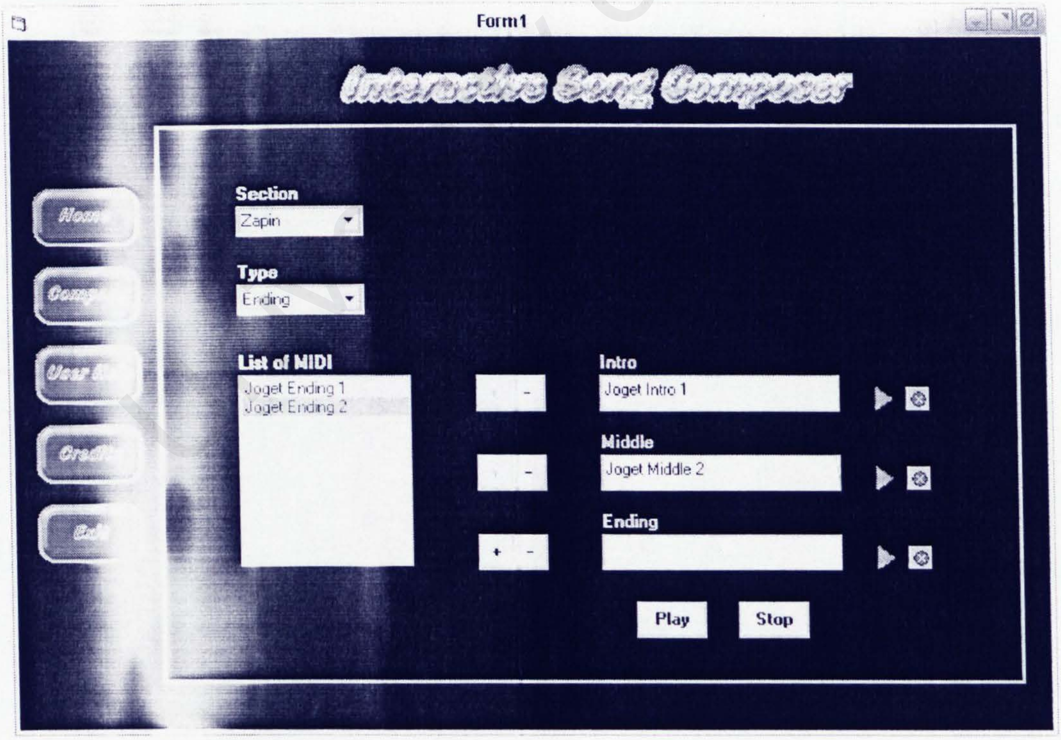
This program is build to help users experience in composing song in the early stage.

Requirements:

Windows XP/2000/NT/98
RAM 128MB (minimum)
Intel Pentium 4 (minimum)

Copyrights © 2005-2006 FSKTM. All rights reserved.

Figure 5.6 Home frame (default frame)


Form1

Interactive Song Composer

Section: Zapin

Type: Ending

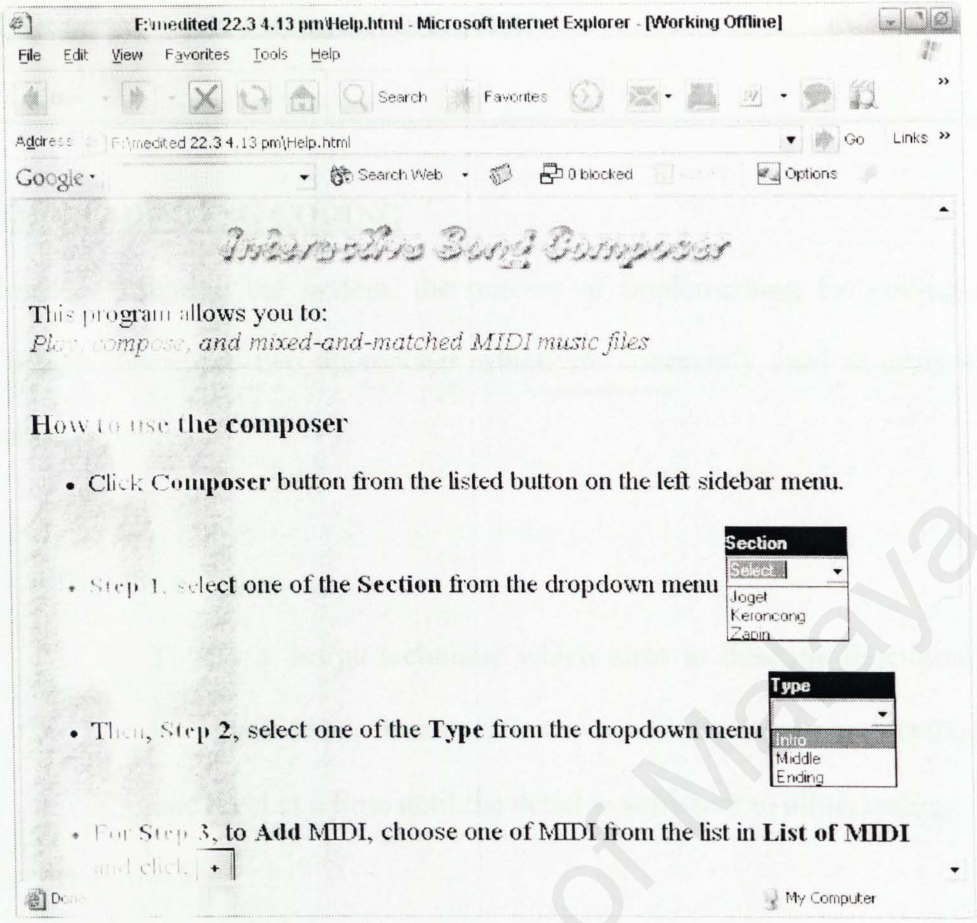
List of MIDI:
Joget Ending 1
Joget Ending 2

Intro: Joget Intro 1

Middle: Joget Middle 2

Ending:

Figure 5.7 Compose frame (main frame)



F:\medited 22.3 4.13 pm\Help.html - Microsoft Internet Explorer - [Working Offline]
File Edit View Favorites Tools Help
Search Favorites
Address F:\medited 22.3 4.13 pm\Help.html
Go Links
Google Search Web 0 blocked Options
Done
My Computer

Interactive Song Composer

This program allows you to:
Play, compose, and mixed-and-matched MIDI music files

How to use the composer

- Click **Composer** button from the listed button on the left sidebar menu.
- Step 1, select one of the **Section** from the dropdown menu
- Then, Step 2, select one of the **Type** from the dropdown menu
- For Step 3, to **Add** MIDI, choose one of MIDI from the list in **List of MIDI** and click +

Figure 5.8 User Manual (an html document)

6.1 IMPLEMENTING CODING

After finished designing the system, the process of implementing the coding for the system begins. There are two approaches which are commonly used in designing and implementing coding:

- Top-down design
  - This is a design technique which aims to describe functionality at a very high level, then partition it repeatedly into more detailed levels one level at a time until the detail is sufficient to allow coding.
- Bottom up
  - The opposite of top-down design. This technique designs from higher levels of abstraction down to lower, specifying sequences of action in increasing detail until the actual code.

As for the ISC, the design approach that had been used is the top-down design approach. This means the development of ISC started by dividing and partitioning the system into smaller units based on their functionality. This approach had been chosen because it makes it easier to understand the whole system by dividing and partitioning it into smaller units. The ISC coding is divided into two main units, which are module and interface/navigation.

6.2 CODING DOCUMENTATION

Program documentation is a set of written descriptions that explain to readers about what the programs do and how they do it. This documentation may categorize as internal documentation whereas the others are known as external documentation. Internal documentation is descriptive material written directly within the code. It contains information directed at someone who will be reading the source code of the program. Comment within the code is the example of internal documentation. It provides line-by-line explanation of what the program is doing such as the description about data structures, algorithms and so on. Besides, it also breaks the code into phases that represent major activities.

Different programming language uses distinguish comment syntax. For instances, in Visual Basic coding, each comment tag is proceeding with a single code ('). Any statement at the same line after a single code will be ignored during execution time. This comment line will appear in green colour.

Example:

```
' This is a comment.
```

Apart from the comment usage, meaningful variable names and statement labels also being used to increase code readability.

6.3 CODING IMPLEMENTATION

6.3.1 Module

Below shown the codes that are included in the module:

Declaration for supports/expanding MIDI types

```
Declare Function mciSendString Lib "winmm.dll" _
   Alias "mciSendStringA" _
   (ByVal lpstrCommand As String, _
   ByVal lpstrReturnString As String, _
   ByVal uReturnLength As Long, _
   ByVal hwndCallback As Long) As Long
Declare Function mciGetErrorString Lib "winmm.dll"_
   Alias "mciGetErrorStringA" _
   (ByVal dwError As Long, _
   ByVal lpstrBuffer As String, _
   ByVal uLength As Long) As Long
Declare Function GetShortPathName Lib "kernel32" _
   Alias "GetShortPathNameA" _
   (ByVal lpszLongPath As String, _
   ByVal lpszShortPath As String, _
   ByVal cchBuffer As Long) As Long
```

Set the database path and connection

```
Sub hubung()

    Dim appath As String

    appath = "D:\modified 22.3 4.13 pm"

    b = appath & "\data\compose_97.mdb" ' location db

    Set db = OpenDatabase(b)

End Sub
```

To play the MIDI file

```
Public Sub OpenMidi()

   Dim sFile As String

   Dim sShortFile As String * 67

   Dim lResult As Long

   Dim sError As String * 255

   Dim appath As String

   appath = "D:\modified 22.3 4.13 pm\midi"

   'Set the path and filename to open.  I am using the ronggeng.mid

   'I just copied it to this projects folder.

   sFile = appath & "\ronggeng.mid" ' the path for the specified MIDI


   'make another API call to get the short

   'filename version.

   lResult = GetShortPathName(sFile, sShortFile, _

                        Len(sShortFile))
```

```
    sFile = Left$(sShortFile, lResult)

    'Make the call to open the midi file and assign
    'it an alias
    lResult = mciSendString("open " & sFile & _
    " type sequencer alias combine", ByVal 0&, 0, 0)

    'Check to see if there was an error
    If lResult Then
        lResult = mciGetErrorString(lResult, sError, 255)
        Debug.Print "open: " & sError
    End If
End Sub

Public Sub PlayMidi()
    Dim lResult As Integer
    Dim sError As String * 255

    'Make the call to start playing the midi
    lResult = mciSendString("play combine", ByVal 0&, 0, 0)

    'Check to see if there were any errors
    If lResult Then
```

```
        lResult = mciGetErrorString(lResult, sError, 255)

        Debug.Print "play: " & sError

    End If

End Sub
```

To stop the now playing MIDI

```
Public Sub CloseMidi()

    Dim lResult As Integer

    Dim sError As String * 255

    'Make the call to close the midi file


    lResult = mciSendString("close combine", "", 0&, 0&)


    'Check to see if there were any errors

    If lResult Then

        lResult = mciGetErrorString(lResult, sError, 255)

        Debug.Print "stop: " & sError

    End If


End Sub
```

6.3.2 Interface/Navigation

<u>To display the splash screen</u>

```
Private Sub Form_Load()
   tmrCountDown.Enabled = True

   'Set countdown timer to 10 seconds.
   timeInterval = 10
   msg = timeInterval

   'Progress Bar
   lblProgBar.Width = 0
   lblProgBar.Visible = False
   lblContainer.Visible = False
End Sub

Private Sub Label2_Click()
'do nothing
End Sub

Private Sub tmrCountDown_Timer()
   Dim i As Integer
   Dim sCDown As String

   lblContainer.Visible = True
   lblProgBar.Visible = True

   sCDown = Right(lblCountDownTimer.Caption, 1)

   'Set i to 0 making sure it decreases in
   'steps of -1
```

```
  i = 0

  If sCDown = "0" Then
    tmrCountDown.Enabled = False
    frmMain.Show
    Unload Me
    Set frmSplashDemo = Nothing
  ElseIf i < timeInterval Then
    i = i + 1
    msg = msg - 1
    lblCountDownTimer.Caption = "Seconds remaining: " & msg

    lblProgBar.Width = lblProgBar.Width + 320
  End If

End Sub
```

On form load

```
Private Sub Form_Load() 'start application
  Call hubungi_db
  Dim SQL1 As String
  SQL1 = "SELECT * FROM table_kategori"
  Set rs1 = db.OpenRecordset(SQL1)

  While Not rs1.EOF
    cbokategori.AddItem rs1!kategori
    rs1.MoveNext
  Wend

  rs1.MoveFirst

  fracompose.Visible = False

  cmdAdd_intro.Enabled = False

  cmdAdd_middle.Enabled = False

  cmdAdd_end.Enabled = False

  imgbanner.Visible = False

End Sub
```

Action perform when Home and Compose is being clicked

```
Private Sub btnCompose_Click()

  frahome.Visible = False

  fracompose.Visible = True

  imgbanner.Visible = True

End Sub


Private Sub btnHome_Click()

  frahome.Visible = True

  fracompose.Visible = False

  imgbanner.Visible = False

End Sub
```

To select section (intri/middle/ending)

```
Private Sub cbobahagian_Click()

Call hubungi_db

Dim SQL1 As String

Dim strintro1 As String

Dim strintro2 As String

strintro1 = cbokategori 'pilihan kategori (jenis music)

strintro2 = cbobahagian 'pilihan bahagian intro, middle or ending


SQL1 = "SELECT * FROM " & strintro2 & " WHERE kategori = '" & strintro1 & "' "

Set rs1 = db.OpenRecordset(SQL1)


 lstsenarai.Clear



  While Not rs1.EOF


    lstsenarai.AddItem rs1!senarai

    rs1.MoveNext

   Wend


  rs1.Close



  If strintro2 = "Intro" Then
```

```
cmdAdd_intro.Enabled = True
cmdAdd_middle.Enabled = False
cmdAdd_end.Enabled = False


Else
If strintro2 = "Middle" Then

cmdAdd_intro.Enabled = False
cmdAdd_middle.Enabled = True
cmdAdd_end.Enabled = False


Else
If strintro2 = "Ending" Then

cmdAdd_intro.Enabled = False
cmdAdd_middle.Enabled = False
cmdAdd_end.Enabled = True
```

```
  End If

  End If

  End If

End Sub

Private Sub cbobahagian_DropDown()

'combo box untuk pilih bahagian intro, middle atau ending

 Call hubungi_dbsong

  Dim SQL1 As String

  SQL1 = "SELECT * FROM table_part"

   Set rs1 = db.OpenRecordset(SQL1)

   rs1.MoveFirst

  cbobahagian.Clear

  While Not rs1.EOF

     cbobahagian.AddItem rs1!part

     rs1.MoveNext

  Wend

End Sub
```

To place selected MIDI filename into the textbox provided

```
Private Sub cmdAdd_intro_Click() 'button add intro

Dim strintro1 As String

strintro1 = lstsenarai

TextIntro.Text = lstsenarai.Text

End Sub


Private Sub cmdAdd_middle_Click() 'button add middle

Dim strmiddle1 As String

strmiddle1 = lstsenarai

TextMiddle.Text = lstsenarai.Text

End Sub


Private Sub cmdAdd_end_Click() 'button add ending

Dim strend1 As String

strend1 = lstsenarai

TextEnding.Text = lstsenarai.Text

End Sub
```

To open, play and stop current playing MIDI

```
Private Sub Cmdplay_Click() 'button play bawah
   OpenMidi
   PlayMidi
End Sub

Private Sub Cmdstop_Click() ' button stop bawah
   CloseMidi
End Sub
```

6.4 SYSTEM DEVELOPMENT ENVIRONMENT

Before going further on the system implementation and development, it is necessary to look at the development environment of the ISC. The development environment of the ISC is divided into two categories, which are:

- Hardware
- Software

6.4.1 Hardware

As for the hardware, the development of ISC was implemented on a computer with the following specifications:

- Processor	: AMD Athlon XP Barton 2600+
- RAM	: 512MB
- Hard Disk Drive	: SATA120GB

6.4.2 Software

The software that had been used throughout the development phase of IST is listed below:

- Operating System : Windows XP Professional SP2
- Development Tool : Microsoft Visual Basic 6.0
  Microsoft Access

6.5 SUMMARY

During system implementation, system requirements and designs were converted into program codes. Besides, it also involves development environment setting such as the operating system and the driver installation. Several software tools were used to deploy the design into machine-readable language and then in turn to produce the required applications.

CHAPTER 7: SYSTEM TESTING

7.1 INTRODUCTION

In the process of developing software, software-testing process plays an important role. This should be done in various aspects of the software. Software testing attempts to find all the faults in programs, as it being developed and even when it is presumed to be finished. Testing and often be ineffective in finding all of programs faults, but it is still an essential part of the software process. As program become more complex testing become more important to verify program functionality under all conditions.

Once the entire system is validates, it must be combined with other system elements such as software, hardware, end-user and etc. System testing verifies that elements are functioning properly and overall system performance and objectives are archived. Based on this, we can get some information and ideas in order to come up with better software. The resulting information from the survey is summarized and forms the reminder of this chapter.

7.2 OBJECTIVE OF TESTING PHASE

The objective of testing in the process of development of this system is to test the program, whether it really works or not. Various stage of testing was carried out at various stage of development. Testing is done to ensure the quality of the software system in term of design errors, technical errors, coding, implementation error and etc. The objectives of testing are as below:

- **Identify Errors**

  Inspecting the function of each part of the system is very important. This can help us to identify errors. Once errors are found, proper action should be taken to overcome this problem.

- **Removing errors**

  This process involves fixing the problem found during the list phase above. The actual process of solving problems will vary with different problems.

7.3 TYPES OF TESTING

There are seven basic types of software test. There are unit/module test, integration test, external function test, regression test, system test, acceptance test and installation test. Testing is actually series of different test whose primary purpose is fully exercising the computer based system. Although each test has different purpose, but all work to verify the system elements have been properly integrated and perform allocated.

For this Interactive Song Composer, three type testing have been performed. There are:

1. Unit/Module Testing
2. Integration Testing
3. Overall System Testing

Testing strategies used during the development of this application consists of the testing as show as above.

7.4 UNIT/MODULE TESTING

Unit testing focuses on verification effort on the smallest unit of software design. The unit test is write-box oriented and the step can be conducted in parallel for multiple components. In unit testing, we examine functions and sub-routines were tested to check for coding or logical error. The basic steps stated below were followed during the unit testing.

For this Interactive Song Composer testing was done during coding phase. After the source code of the module have been developed, reviewed and verified for the correct syntax unit testing case was designed. The module was tested to ensure that is operates correctly.

The following areas are tested during unit testing.

- **Module testing**

  In this Interactive Song Composer, there are all 4 modules which are main, compose, credits and user manual. Each of these modules has to be tested carefully to avoid any problem during deliverable phase.

Button and icons are tested to check that they are link to the correct path. Types of buttons and icons are also checked so that the same type of buttons and icons is used for the same modules.

This module can be tested from different aspects which are as below.

- **Interface**

  Make sure all the buttons and icons are well organized; each interface of the module is tested

- All possible independent program path are executed

  Ensures that the control structures are implemented correctly

- **Boundary value analysis**

  Ensure that the module operates property at boundaries established to limited/restricted processing

- **Error handling path**

  Ensure that the specific module executes the recovering process when an error occurs

7.5 INTEGRATION TESTING

Once individual components are tested properly to ensure they work independently, these modules will be integrated and tested as a whole. Testing is tested between different modules. Buttons and icons are checks tat it able to link from module to module without any abrupt stop.

Incremental integration approach was applied during the development of Interactive Song Composer. Interactive Song Composer was constructed and tested in small arguments, where errors were easier to isolate and correct. Error will be corrected before proceeding to the next integration.

Top-down integration testing is an incremental approach to construct the system structure.

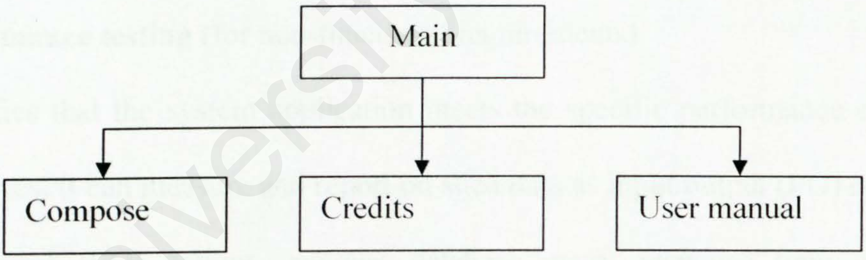


Figure 7.1: Top-down integration testing for Interactive Song Composer

Referring to the figure above, the component module will be integrated first with the main module then followed by credits and then user manual module.

7.6 SYSTEM TESTING

System testing is to ensure that the system is functioning well under a larger system.

There are several steps to perform the test on Interactive Song Composer .There are:

- **Functional Testing** (for functional requirements)

  Used to assess whether the application does what ever it suppose to do in accordance with project's requirements. The guidelines use for testing Interactive Song Composer is:

  - Have a high probability of detecting a fault
  - Know the expected actions and output
  - Test both invalid and valid data
  - Never modify the system just to make testing easier
  - Have stopping criteria

- **Performance testing** (for non-functional requirements)

  It verifies that the system application meets the specific performance efficiency objectives. It can measure and report on such data as input/output (I/O) rates, total number of (I/O) actions, average database query response time and CPU utilization rates.

  Interactive Song Composer was tested on the respond time of database query and sound play on the speaker.

- **Acceptance Testing**

  Enable the user to determine whether the application meets their requirements

- **Regression Testing**

  Determine whether any errors have been introduced during error-fixing process

- **Usability Testing**

  Verifies that the system is easy to be use and user interface appearance is appealing and the system is user friendly. Make sure all the buttons and icon are understandable by user and make sure that user will not lost with the navigation

- **User Acceptance test**

  User acceptance test is the test that is conducted by the users of the system. This test will enable the users to give their feedbacks on the system. These feedbacks are vital in providing the developer with ideas to enhance the system before it is made available to the real users.

CHAPTER 8: SYSTEM EVALUATION

8.1 INTRODUCTION

Evaluation process is done to the 'Interactive Song Composer' right after it undergoes the system testing. Evaluation in this context means the process of examining a system or system component to determine the extent to which specified properties are present.

In this chapter, I will discuss the system strength, system constraints and future enhancements.

8.2 SYSTEM STRENGTH

- **Graphical Approach**

  This system used graphical approach to present itself in an interesting environment that will help the user understand the results easier. The system itself developed with simple interface.

- **User Friendly**

  'Interactive Song Composer' is very easy to use. Once the system started, it shows the menu where the users will have the control from main menu. Buttons will navigate the user to each certain process.

- **Educational**

  This system actually could be use learning-aid for music student in their study at the early stage of understanding in composing a song. The learning process will be easier for them and also for non music student that wanting to know how to compose a song basically.

8.3 SYSTEM CONSTRAINTS

- **Lack of Features**

  As this system is a simple song composer system, it does not have many features like other existing song editor or song composer system.

- **No Record**

  The system cannot store or save a record for each MIDI that have been previewed or played by user. The system will use the same database again and again for temporary record before the system can be run.

8.4 FUTURE ENHANCEMENT

- **Enable Save Function**

  System can save the combination of the selected MIDI by user selects in user directory and not in system directory only. This function is made so that in the future user doesn't need to repeat the same procedures to produce the same outcome or the same combination of MIDI. So every saved composed song will be stored in a different record or in user directory.

8.5 PROJECT LIMITATION

There are several limitations in this proposed system and those limitations are listed below:

- This application only can be run in Windows operating system.
- The application just allow user to select the combination of MIDI as their wish but could not save it later for the user's collection for song that have been composed by user.
- This application is only provided in one language which is English.
- Focused on beginner's user.
- Contains 2D environment.

8.6 PROBLEM ENCOUNTERED

➢ Lack of Source/References

It is very difficult to develop and implement the system. It is because the coding is involving a MCI module which is use to support sound and it's hard to find a source code or existing system for us to take it as reference besides it's quite hard to understand the functionality of every coding statement.

➢ Difficulties During Project Study and Requirement Analysis

This system involves music fields and education. Therefore, basic knowledge is needed as a foundation in building an application of this nature.

- Kamran Iqbal, Tony Jamieson, Developed with Training Associates, INC. (1999). *Microsoft Visual Basic 6.0 Fundamentals.* Redmond, Washington: Microsoft Press.
- Richard Mansfield (1999). *Visual Basic 6: Database Programming for DUMMIES.* Indianapolis, Indiana: Wiley Publishing, INC.
- Julia Case Bradley, Anita C. Millspaugh (2002). *Programming in Visual Basic 6.0.* USA: McGraw-Hill/Irwin.
- Joey F.George, Dinesh Batra, Josep S. Valacich, Jeffrey A. Hoffer (2004). *Object-Oriented Systems Analysis and Design.* New Jersey: Pearson Prentice Hall.
- J.L.Whitten, L.D.Bentley, K.Ditman (2004) *System Analysis and Design Method Sixth edition*, McGraw-Hill.
- Supporting Musical Composition by Externalizing the Composer's Mental Space by Shigeki Amitani and Koichi Hori
- Sound Compositions for Expanding Musicianship Education by Thomas A. Regelski
- Computer-Music Interfaces: A Survey by Bruce W. Pennycook

REFERENCES

Internet Resources

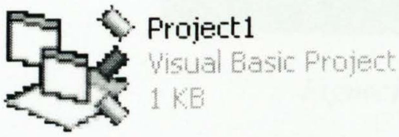
1. http://www.p2p.wrox.com
2. http://www.visualbasicforum.com
3. http://www.devx.com/vb2themax/Tip
4. http://www.musicmall-asia.com/malaysia/syncretic.html
5. http://www.databasedev.co.uk/table_design_tips.html
6. http://www.keybdwizrd.com/music.html
7. http://www.geocities.com/myril88/iramakita.html

Running the application

The steps are as follows:

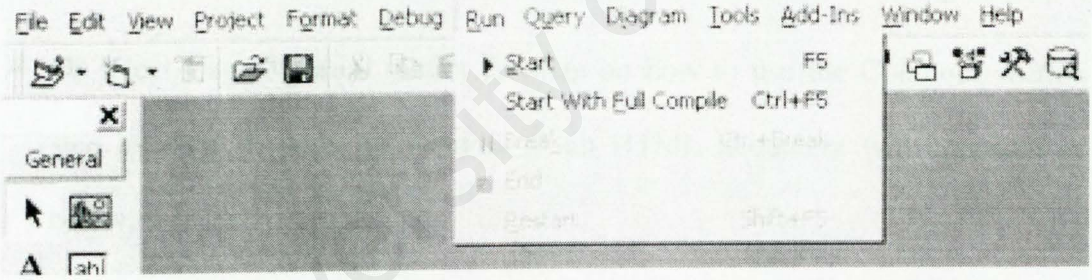
Step 1:

➢ Launch a file name Project1



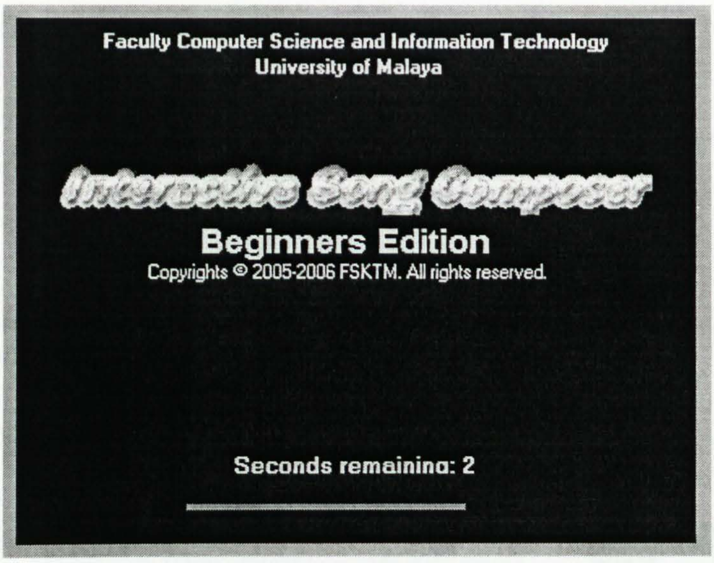
Step 2:

➢ After the application has been loaded, select **Run** from the Menu Bar and choose **Start** to run the application


File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
Start F5
Start With Full Compile Ctrl+F5
Break Ctrl+Break
End
Restart Shift+F5
General

Step 3:

➢ A Splash Screen will appeared, the application will be loaded right after the Splash Screen finished.



Faculty Computer Science and Information Technology
University of Malaya

Interactive Song Composer

Beginners Edition

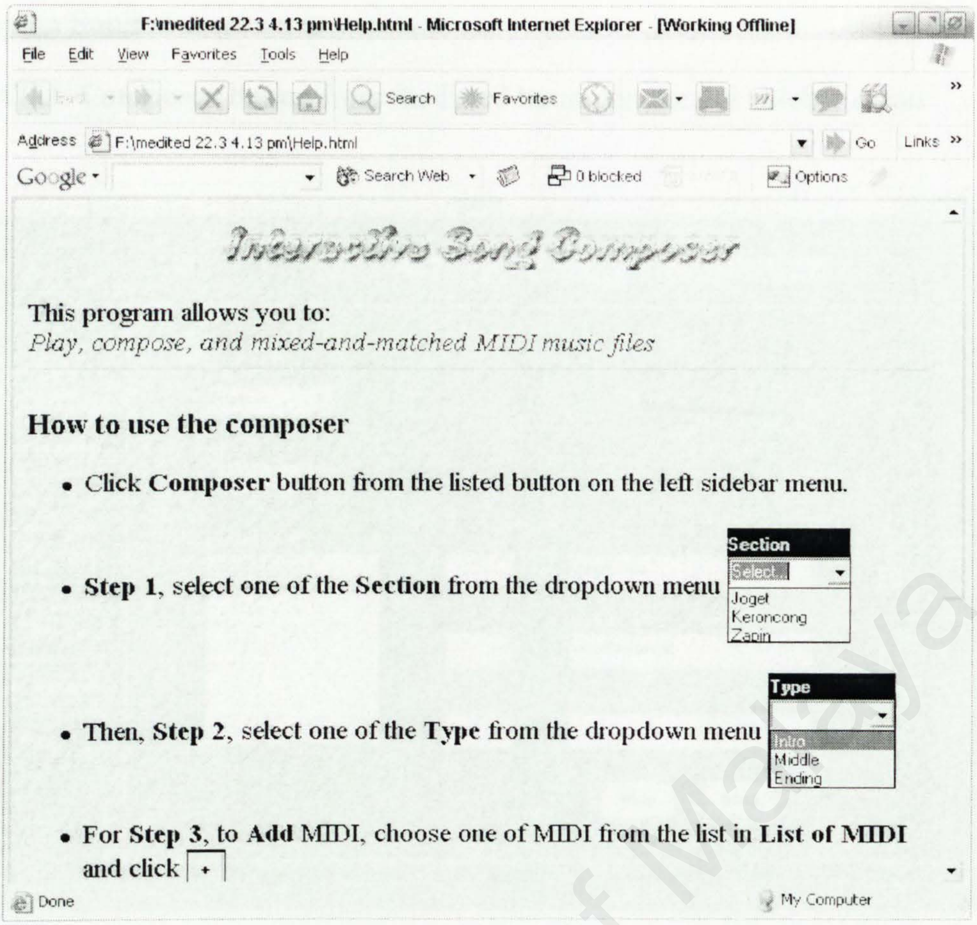
Copyrights © 2005-2006 FSKTM. All rights reserved.

Seconds remainina: 2

Figure1.0 Splash Screen

Step 4:

- Choose **Compose** button the left side of the application to start compose
- Or select **Home** button to view acknowledgement for the system in generally
- Or select **User Manual** button for help on how to use the Compose section step by step. A pops up window of an HTML document will appeared as below.


F:\medited 22.3 4.13 pm\Help.html - Microsoft Internet Explorer - [Working Offline]
File Edit View Favorites Tools Help
Back Search Favorites
Address F:\medited 22.3 4.13 pm\Help.html Go Links
Google Search Web 0 blocked Options
Section
Select...
Joget
Keroncong
Zapin
Type
Intro
Middle
Ending
Done
My Computer

Interactive Song Composer

This program allows you to:
Play, compose, and mixed-and-matched MIDI music files

How to use the composer

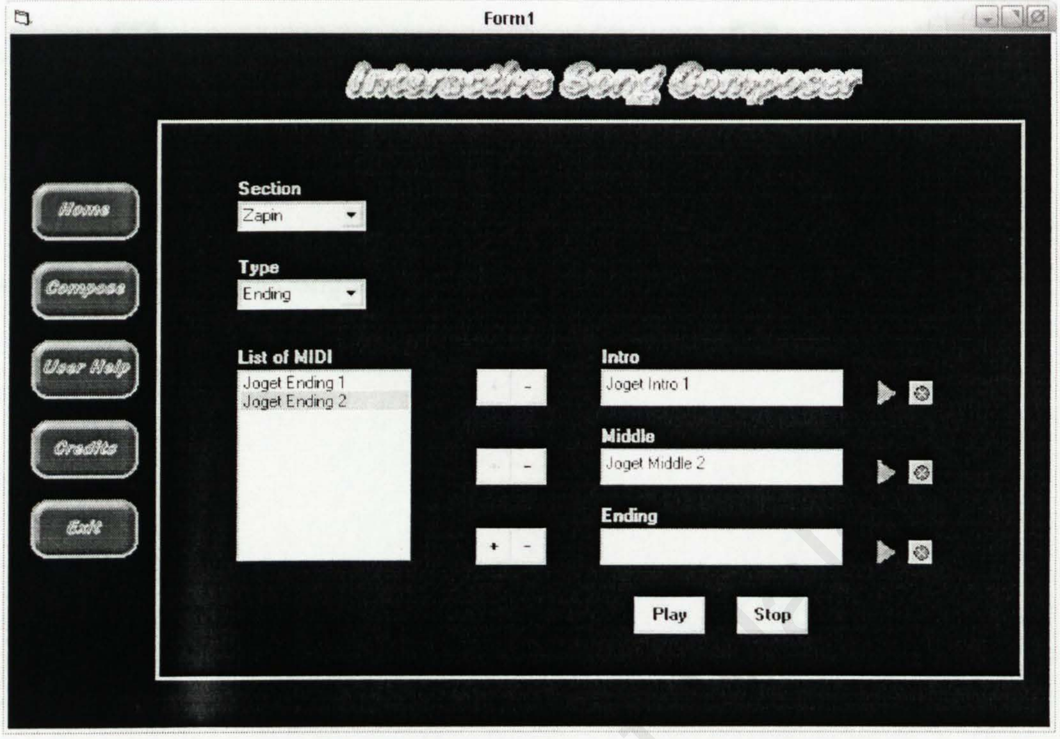
- Click **Composer** button from the listed button on the left sidebar menu.
- **Step 1**, select one of the **Section** from the dropdown menu
- Then, **Step 2**, select one of the **Type** from the dropdown menu
- For **Step 3**, to **Add** MIDI, choose one of MIDI from the list in **List of MIDI** and click +

Figure 2.0 User Manual (an html document)

➢ Or select **Credits** button to get to know who have been involved during the development of this project

Composing a Song

- Click **Composer** button from the listed button on the left sidebar menu.



Section: Zapin

Type: Ending

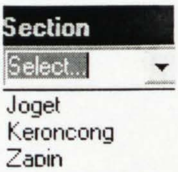
List of MIDI:
Joget Ending 1
Joget Ending 2

Intro: Joget Intro 1

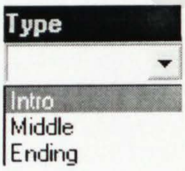
Middle: Joget Middle 2

Ending:

- **Step 1**, select one of the **Section** from the dropdown menu



- Then, **Step 2**, select one of the **Type** from the dropdown menu



- For **Step 3**, to **Add** MIDI, choose one of MIDI from the list in **List of MIDI** and click +

- Click ▶ for preview of the selected MIDI and click ⏹ to stop the preview.

- To **Remove** or **Undo** the selected MIDI, click -

- After finished selecting MIDI for the 3 section: **Intro**, **Middle** and **Ending**, click button **Play** to play the combination of the 3 MIDI.