



**LANCASTER UNIVERSITY
FACULTY APPLIED SCIENCE
COMPUTING DEPARTMENT**

**Investigations into GUI Builder and their Effects on
GUI Design**

**MSc in Distributed Interactive Systems
Dissertation Report**

**By:
Masitah Ghazali**

**Supervisors:
Prof. Alan Dix
Dr. Ruth Stalker**

**Date of submission:
September 2002**

Online Documentation:
<http://www.lancs.ac.uk/postgrad/ghazalim/index.htm>

I certify that the material contained in this dissertation is my own work and does not contain significant portions of unreferenced or unacknowledged material. I also warrant that the above statement applies to the implementation of the project and all associated documentation.

In the case of electronically submitted work, I also consent to this work being stored electronically and copied for assessment purposes, including the department's use of plagiarism detection systems in order to check the integrity of assessed work.

I agree to my dissertation being placed in the public domain, with my name explicitly included as the author of the work.

Date:

Signed:

Abstract

The usage of user interface software tools, particularly the graphical user interface (GUI) builders, has been increasing from year to year, mainly in the industrial field. The commercial and research tools/builders have proven to be either successful or less successful in supporting the users during the implementation. This report presents a proposal for a GUI builder's project. This involves the study of their effects on GUI design in order to discover the better way of providing a generic GUI builder. The primary aim of this project is to suggest ways of providing the 'better' more generic GUI builder. The obtained results, i.e. the generic guidelines, would be the significant contribution part of this project. The prototype is implemented in such a way that it illustrates the realisation of generic guidelines onto a GUI builder from the usability context, and not a fully functional prototype.

Acknowledgements

I thank Prof. Alan Dix for guiding me in the right direction and for his useful advice.

My thanks also go to Dr. Ruth Stalker, who has been very helpful from the beginning of this project and for always believing in me.

I thank all staff at the Department of Computing, Lancaster University for their assistance.

I thank my mom and dad and family members for their love and encouragement.

I thank my many friends for their continuous support.

I give thanks to God the Almighty for giving me strength and energy in completing this dissertation.

Table of Contents:

Chapter 1: Introduction	1
1.1 Graphical User Interface (GUI) Builders.....	1
1.2 Objectives of Dissertation.....	2
1.3 Dissertation Structure.....	2
Chapter 2: Literature Review	4
2.1 Graphical User Interface (GUI).....	4
2.2 Graphical User Interface (GUI) Builders.....	5
2.2.1 Design Tools.....	5
2.2.2 Software Engineering Tools.....	6
2.2.3 Language-based Tools.....	6
2.2.4 Application Frameworks.....	7
2.2.5 Model-based Generation.....	7
2.2.6 Interactive Tools.....	7
2.3 Interface Designs.....	8
2.3.1 The Eight Golden Rules.....	8
2.3.2 Interactiveness of GUIs.....	9
2.3 Conclusion.....	13
Chapter 3: Analysis on Several GUI Builders/Generators	14
3.1 JGUIDev 1.0 and JGUIDev 1.1.....	14
3.2 Qt Designer Trolltech.....	16
3.3 wxWindows Dialog Editor.....	18
3.4 Visaj.....	20
3.5 lxb.....	21
3.6 Conclusion.....	22
Chapter 4: Survey on lxb Software	24
4.1 Web-based User Interface Evaluation with Questionnaire.....	24
4.2 QUIS Results.....	25
4.2.1 Overall Reaction to the Software.....	25
4.2.2 Screen.....	26
4.2.3 Terminology and System Information.....	26
4.2.4 Learning.....	27
4.2.5 System Capabilities.....	27
4.3 CSUQ Results.....	28
4.3.1 Easiness.....	28
4.3.2 Completion of Work.....	29
4.3.3 Users' Satisfactory.....	29
4.3.4 Error.....	29
4.3.5 Information.....	29
4.3.6 Interface.....	30
4.3.7 lxb as a whole.....	30
4.4 Negative and Positive Aspects of lxb resulted from QUIS and CSUQ.....	31
4.4.1 Negative Aspects.....	31
4.4.2 Positive Aspects.....	33
4.5 Conclusion.....	35
Chapter 5: Generic Guidelines	37
5.1 Derivation Steps.....	37
5.2 Brief Descriptions on Criteria.....	37
5.2.1 Multiple Windows.....	38
5.2.2 Consistency.....	39

5.2.3 Meaningful Feedback.....	40
5.2.4 Online Help.....	41
5.2.5 Target the User.....	41
5.2.6 Accuracy.....	42
5.2.7 Cut and copy, drag and drop techniques.....	42
5.2.8 Shortcuts.....	42
5.2.9 Preview.....	43
5.2.10 Hierarchy.....	43
5.3 Generic Guidelines.....	43
5.4 Conclusion.....	45
Chapter 6: Design.....	46
6.1 Direct Manipulation.....	46
6.2 WIMP Interface.....	47
6.3 State Transition Network (STN).....	47
6.4 Screen Design.....	50
6.5 Conclusion.....	51
Chapter 7: Implementation.....	52
7.1 Technology Used.....	52
7.2 The Prototype.....	53
7.2.1 Screen Interface of the GUI builder.....	53
7.2.2 Drag and Drop.....	55
7.2.3 Feedback.....	56
7.2.4 Help Documentation.....	57
7.3 Conclusion.....	58
Chapter 8: Further Work.....	59
8.1 Improvements.....	59
8.2 Extension Work.....	60
8.3 Future Research.....	61
Chapter 9: Conclusion.....	64
References & Bibliography.....	66
Appendixes	
Appendix A: Questionnaire for User Interface Satisfaction Results (a)	i
Appendix B: Questionnaire for User Interface Satisfaction Results (b)	ii
Appendix C: QUIS Results Analysis	iv
Appendix D: Computer System Usability Questionnaire Results (a)	viii
Appendix E: Computer System Usability Questionnaire Results (b)	ix
Appendix F: Analysis of CSUQ results	xi
Appendix G: Negative and Positive Aspects of QUIS and CSUQ	xiv
Appendix H: Prototype Implementation Check List	xviii

Table of Figures:

Figure 2.1: Typical Graphical User Interface Systems [6].....	4
Figure 3.1: JGUIDev Screen Interface.....	15
Figure 3.2: JGUIDev Error Message.....	16
Figure 3.3: Qt Designer Screen Interface.....	17
Figure 3.4: wxWindows Dialog Editor Screen Interface.....	19
Figure 3.5: Visaj Screen Interface.....	21
Figure 3.6: lxb Screen Interface.....	22
Figure 5.1: Steps taken to produce Generic Guidelines.....	37
Figure 6.1: The General Interaction Framework.....	47
Figure 6.2: State Transition Network for Designing Dialogue.....	49
Figure 6.3: The Proposed Screen Design.....	50
Figure 6.4: Cross Grid.....	51
Figure 6.5: Dots Grid.....	51
Figure 7.1: Prototype Screen Interface.....	53
Figure 7.2: Toolbar and Generate Icon.....	54
Figure 7.3: Widgets.....	55
Figure 7.4: Drag and Drop.....	56
Figure 7.5: Example of Dialogue to be saved.....	57
Figure 7.6: Feedback Message.....	57
Figure 7.7: Help Documentation.....	58
Figure 8.1: Research Plan.....	63

Chapter 1: Introduction

1.1 Graphical User Interface (GUI) Builders

Today, there is enormous number of software tools that have been invented by the researchers, as well as by those who are from the commercial area. It is believed that in the UNIX market alone, over US\$133 million of tools were sold in 1993, which is about 50,000 licenses [17].

In today's world, the GUI software tools are widely used in order to assist the end user and the designer themselves to create the user interfaces. Direct manipulation interfaces have been involved in many development of GUI, as they offer the programmer ways to deal with elaborate graphics, multiple ways for giving the same command, multiple asynchronous input devices, a "mode free" interface, where the user can give any command at virtually any time, and rapid "semantic feedback", where determining the appropriate response to user actions requires specialised information about objects in the program [9].

Due to the fact that there are so many tools out there, they sometimes create difficulties and complications to the users when utilising the tools. Therefore, one way to improve, if not to overcome this problem, is by suggesting the 'better' more generic GUI builder. This project takes the approach of producing generic guidelines from the design context.

The significant point of having it from the design context is due to the nature of the *generic* term itself. Anything that is generic has the probability and chance to be out grown or go beyond the scope (that this is set and dependable on the designer), if the generic term used doesn't have something to govern them. Thus, alongside the investigations into GUI builders, this project also analyses the effects of GUI builders on GUI design.

Some definitions are worth describing in order to assist the understanding of some terms, in this report in particular, and throughout the project in general. The user interface (UI) of a computer program is the part that handles the output to the display and the input from the person using the program. The term user interface tools have been called various names over the years, and that includes “*Toolkits* and *Interface Builders*” [9]. The terms *GUI builder* and *software tool* will be used for all software aimed to help create the graphical user interface, particularly in creating the design dialogues. These terms will be used interchangeably in this report.

1.2 Objectives of Dissertation

The focus of this project is to investigate the various GUI builders/generators, from both the research and commercial areas. Further inspection is carried out on the results of the analysis in order to derive to the generic guidelines from the design context. The objectives have been divided into two; primary and optional. The objectives of this project are:

Primary Aims

- To perform a literature survey on GUIs and GUI builders and good UI design
- To look at various GUI builders and generators and establish what makes a good GUI builder
- To test and evaluate one GUI builder to see its usefulness and to what extent the GUI constrains design (for better or worse) by using a group of 70 users
- To suggest ways of providing ‘better’ more generic GUI builders

Optional (yet required) Aim

- To design and develop a prototype which illustrates all findings, which this is purposely designed and implemented in the context of usability, and not in the form of a fully functional GUI builder tool

1.3 Dissertation Structure

Chapter 1 is the introduction to the Graphical User Interface builders in general. It also consists of the objectives of the dissertation, which this encompasses the primary and essential objectives that are required towards the completion of the project.

Chapter 2 consists of the literature reviews that also acts as the background research prior to the investigation of the project.

Chapter 3 briefly explains the analysis of the five selected graphical user interface software tools.

Chapter 4 describes the survey that has been carried out on lxb software tool, and the analysis of the survey's results.

Chapter 5 is the generic guidelines derived from the analysis to enlighten the graphical user interface builder developers the proposed generic GUI builder. This contribution is the highlight of this project.

Chapter 6 explains the design steps that have been taken to develop a prototype of a generic graphical user interface builder that illustrates the guidelines from Chapter 5.

Chapter 7 focuses on the implementation of the prototype from the usability perspective.

Chapter 8 elaborates the suggestions for further work regarding this project.

Chapter 9 summarises the generic guidelines of the graphical user interface in general and the conclusion of the project and dissertation, including the overview of the project. This also includes the reflection towards the problems that have been facing while completing the project.

Chapter 2: Literature Review

The literature survey is carried out as to fulfil the first objective of the project, that is to gathering as much information about GUIs, GUI builders and good UI design. This chapter, however, will emphasise more on the areas of GUI builders and UI design. The reason being is, both are reckoned to be the problematic domains that need to be carefully identified, as their roles in contributing to the production of the generic guidelines.

2.1 Graphical User Interface (GUI)

Graphical user interface brings different definitions if it is interpreted in a different context. We could see the differences from the following instances.

The first example tries to define GUI from the system application context. In which by quoting from Meyyapan [8], “Graphical User Interface (GUI) is a graphical software layer exists between the human and the system, to provide an easier way to communicate with the back end system for administration, configuration, information exchange, etc”. For a typical GUI system, the GUIs may created from the implementation of Visual C++ or Visual Basic of the Windows platform, Java or UNIX platforms, or HTML or web, which are strongly coupled with the back end system. The figure below illustrates this.

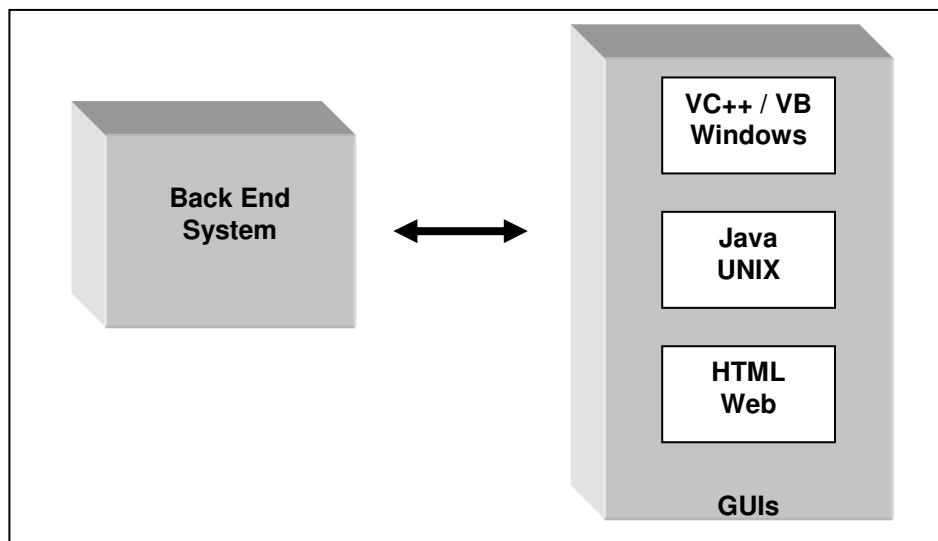


Figure 2.1: Typical Graphical User Interface System [8]

From a different context, the graphical user interface is defined as the graphical features that are offered by the operating systems that exist today, like Microsoft Windows and Linux. The features include components like window, pointer, icon, scroll bar, button, edit box and many more. This definition is widely perceived by many computer users as oppose to the earlier definition. If the graphical user interface of a system comes with its input devices, the graphical user interface of that system is referred to as “look-and-feel” [16].

The invention of GUI started back in the late year of 1970s. This was originated at the Xerox Palo Alto Research Laboratory before the Apple Company used it in their first Macintosh computers. Microsoft later used the same ideas in their first version of the Windows operating system [15][16].

There are many existing object-oriented tools that help the users to write a graphical user interface in much simpler way. From the programming aspect, GUI elements must first be encapsulated inside a class widget. At this point, the element creates an object instances to provide functionality [16]. The following section elaborates further on tools that facilitate the writing of the graphical user interface.

2.2 Graphical User Interface (GUI) Builders

Before proceeding into detail about GUI builders, we might want to know how does it differ from the interface-builder tools. Interface-builder tools are the software tools that are required in order to complete the user interface from the detail specification [14]. It further consists of design tools and software engineering tools.

2.2.1 Design Tools

Design tools utilised the concept of direct-manipulation that emphasises the convenient and rapid building of on-screen prototypes. Even though this prototype may sometimes not have a full database, help, or other facilities, it enables the users to attain the realistic view of how the interface will function. Examples of tools vary from the Visual Editing Tools like Apple’s HyperCard, to Visual Programming Tools such as Prograph and

LabVIEW, and to Contemporary Visual Development, e.g. Microsoft Visual Basic, Borland Delphi, and Symantec Code.

2.2.2 Software Engineering Tools

Software engineering tools uses the “general-purpose programming language”, such as C and C++ when developing the user interfaces. This normally includes extensive toolkits (the user interface programming libraries), which gives an extensive control and great flexibility creating the interface. Some software tools that come with toolkits include the Microsoft Windows Developer, Apple MacIntosh MacApp, UNIC X-Window and Tcl/Tk. Others include Galaxy, Java and JavaScript.

Fei [6], has described user interface tools as “any software that helps create user interfaces”. This statement is similar to the concept that the interface-builder tools apply from the earlier description. By giving an example of Higher-Level tools as one of the user interface tools, he then introduced the GUI builder tools as “any higher level user interface tool with run-time component that help programmer or end-user build graphical interface”.

Myers [9] clearly distinguishes GUI builders into four kinds of tools. These are the language-based tools, application framework, model-based generation and interactive tools.

2.2.3 Language-based Tools

The language-based tools allow the user interface to be specified in a special-purpose language. There are many forms of languages. These include context-free grammars, state-transition diagrams (STD), declarative languages, event languages, etc. The language is usually used to specify the syntax of the user interface. Several representative tools include the VAPS, which is of a commercial system that uses the STModel, Sassafra, and Cousin and Apollo’s Open-Dialogue, which these both specify the user interfaces in declarative language. The problem with these tools is, the designer must specify a great deal about the placement, format, and design of the user interfaces.

2.2.4 Application Frameworks

The application frameworks are the tools that provide frameworks to guide the programming. One of the tools, MacApp from Apple, provides the application in specific details by letting the programmer to specialise the classes. Meanwhile, for Unidraw and Amulet, which are of research frameworks, are slightly different from the MacApp as they aimed at the graphical applications. Due to Amulet graphical data model, it allows the programmer to use the existing built-in routines without having to change anything. Thus, the methods for subclasses need not to be written by the programmer. Other examples are Edge and TGE.

2.2.5 Model-based Generation

The model-based automatic generation provides a number of selections for the tools to choose, from a much higher-level specification. The choices are the solution of the problem mentioned in the language-based tools. Tools like Mickey, Jade and DON, concentrated on creating menus and dialogue boxes. Jade extends the functionality to edit the generated interface by allowing the designer to use a graphical editor, if it is not good enough. Meanwhile for DON that has the most sophisticated layout mechanisms, takes into account the desired window size, balance, columns, symmetry, groupings, etc. UIDE (User Interface Design Environment) is also of this tool category.

2.2.6 Interactive Tools

The last category is called interactive tools. These tools allow the creation of dialogue boxes, menus, and windows, that are part to be part of a larger user interface. They offer a selection of widgets from a pre-defined library, and place them on the screen using a mouse. Property sheets are used to set the other properties of the widgets.

Normally, there is some support for sequencing, for instance, bringing up sub-dialogues when a particular button is hit. This category very much similar to what we have in the market today. DialogEditor, vu and Gilt are of the examples that come from the research area, whilst NeXT Interface Builder, UIM/X for X are of the commercial area.

There are two main problems associated with this particular tool. Firstly, although the interface builders make laying out the dialogue boxes and menus easier, it is only part of the user interface design. This creates a problem, which these tools provide little guidance towards creating good user interface, since they give designers significant freedom. Secondly, the interactive tools do not offer of any assistance with the contents of the graphic pane for any kind of program that has a graphic area. In addition to that, they cannot handle widgets that change dynamically.

2.3 Interface Designs

Efficiency and effectiveness of GUI builder tools are also relying on the design, which this very much concerned with the user interface design and the interactiveness of GUIs. Therefore, the following sections describe these in detail.

Before proceeding with the concept of interactive GUIs, we first review the general ideas and rules concerning the user interface design that applicable in most interactive systems. This is of importance as it provides a style guides on how to bringing out an interactive and usable interface design.

2.3.1 The Eight Golden Rules

The most renowned guidelines are the Eight Golden Rules of Interface Design by Shneiderman [14]. The principles first emphasise on the importance for one 'system' to strive on consistency. Consistency encompasses subjects like the use of colour, layout, capitalisation, fonts, etc., which should be employed throughout.

Secondly, it is vital to ensure it enables frequent users to use shortcuts, in order to reduce the number of interactions and to increase the pace of interaction. It should also offer informative feedback to the users for every action taken. Apparently, two kinds of feedback have been suggested correspond to type of users. For frequent and minor actions, the response can be modest, whilst for infrequent and major actions, the response should be more substantial.

It is also essential for the design dialogues to yield the closure. For example, the informative feedback at the completion of a group of actions should give operators, namely, the satisfaction of accomplishment and a sense of relief. Besides providing feedback to the users, the system should also offer error prevention and simple error handling as much as possible so that the users cannot make a serious error. The erroneous actions should leave the system state unchanged, or, the system should give instructions about restoring the state.

As much as possible, actions should be reversible, which this relieves anxiety, since the user knows that errors can be undone. The system should also support the internal locus of control to give the operators the sense that they are in charge of the system and that system responds to their actions. Also of importance is to reduce the short-term memory load. Therefore, the displays must be kept simple, multiple page displays are consolidated, window-motion frequency is reduced and sufficient training time is allocated for codes, mnemonics and sequences of actions.

By having these in mind, a seamless integration with the interactive GUIs would be able to alleviate the design process. Interactive GUIs is described at great length in the following section.

2.3.2 Interactiveness of GUIs

The following points are amongst the features that could ensure the GUIs interactivity. They vary from the point of visibility of a display to more abstract concepts [5].

Direct manipulation

Having compared between interaction styles, like command language and natural language, the direct manipulation is more significant as it is easy to learn, and allows users to move rapidly using only a pointing device. In addition, it allows errors to be avoided or easily corrected, and encourages exploration. Furthermore, it generates rapid feedback and immediate results.

Windowing systems

Windowing systems, which adopt one style of direct manipulation, contain mechanisms to help the user to move, resize, scroll and generally manage multiple windows. This particular application enhances user interaction and provides benefits such as:

- The user can use multiple sources on screen at once
- The users are shielded from complicated command languages, and allowed to specify objects and actions by pointing and selecting
- The users are able to move inside the border of a window or even resize it, moves it within the bounds of the screen.

A window may contain interaction mechanism like, icons, menus, scroll bars, buttons, and pointers, but some standard components that appear in most windowing systems are check boxes, sliders, navigational buttons, dialogue boxes, title bars and text fields.

One of the challenges of the windowing system is to provide access to multiple sources of information. The following are some of the given solutions:

- Split strategy, where the display is split to show two or more parts of a document
- Tiling strategy, where once a second window is opened. The first is rearranged automatically, so that, both of them are visible
- Piles-of-tiles, where windows are stacked on top of one another (complete overlap)

The design of the interactive components located inside the window has an important effect on the user. Therefore, the amount of data displayed should be minimised by presenting only what is necessary to the user.

Grouping

Grouping of the displayed information is also essential, as it improves readability and can highlight relationships between different groups of data. The lines of the groups should have equal spacing and the text must be in a conventional upper or lower case, which

makes it easier for reading. Furthermore, there are cases where showing a large number of text lines gives the readers clearer sense of context for each sentence, while permitting simpler reading and scanning of the document.

Menu

Another interactive component is the menu, which offers simple textual descriptions of the available functions. The names of the options menu have to be self-explanatory. The menus, which can be included in window systems, need careful design and can become quite complex. There are four kinds of menus:

- Fixed menus, which remains in place until the option is chosen
- Pull-down menus, which are dragged down from a simple menu (menu bar)
- Pop-up menus, which appear when a user clicks on a particular area of screen, which may be designated by an icon. The menu remains in position until the user instructs it to disappear again
- Walk-through or cascading menus, which display all the options chosen one after the other.

Standardisation

The concept of standardisation of screen display is always been an important subject. The reason being is, it will enable the users to know where to find given piece of information. Thus, it is recommended to have a list of all the available actions to be displayed in the same place, along with the title name of the screen or window, always. All buttons and menus should be labelled with meaningful names for the corresponding actions that they perform. In addition, an instructing message should be displayed in places where the possible user action is not clearly defined by the button or action label.

The standardisation is also very much concerned with graphical representations, style of communication and interaction speed.

Graphical representations

Graphical representations play an important role in conveying the information. Graphics entail line diagrams, solid shapes, two or three dimensional, monochrome or multi-coloured. Amongst the information that should be displayed in a graphical way includes the data that required the user to make visual judgements, and the information that involves complex data structures, numeric data, or object-oriented representations.

Communication style

Communication style between the user and the application is another issue in the GUI interactivity. Dialogue boxes are very essential as they are used for exchanging instructions and information that takes place between the user and the computer system, as well as, for confirming the user action, so that the user can easily recover from errors. They are also concerned with how a system links sequences of operations together and maps the representations used on to its functions, which are the operational aspects.

Rapid response

Rapid response, which concerns with the speed of the interactive processes, is always be one of the important factors as it effects the system's usability and efficiency. The results from the user's actions should be displayed immediately on the screen. The rapid response reduces the need for additional actions from the user and makes its navigation through the application more quickly and enjoyable.

Colour

Colour usage should be used conservatively as it plays an important role in the graphical representation and in the screen display. Thus, the graphics, colours and texts should be balanced so that the user interface will not be misleading, unpleasant or unclear to the user.

2.3 Conclusion:

The areas covered in this review seem to be very broad and wide, yet, there are of importance in order to ensure the basic root of the problem is understood and the overall

view of the domain subject is crystal clear. Some significant points that are worth highlighted are, first, the application of direct manipulation concept appears in almost all GUI builders and second, the interactive GUIs criteria have the potential to bring the best out in any interactive systems. The next chapter focuses on the analysis of the GUI builders/generators in order to find what makes a good GUI builder.

Chapter 3: Analysis on Several GUI Builders/Generators

This chapter describes a thorough analysis that had been carried out on five GUI builders and generators. The aim of this chapter is to grasp the concept and criteria of what really makes a good and efficient GUI builders. The selection of software tools must be of the commercial and the research areas.

Having known about the Amulet 3.0, created by Brad A. Myers [9], we have considered in the beginning to take this ‘research’ tool as part of our analysis. Unfortunately, due to some difficulties that have arisen when executing the software, we revoked the decision to drop out this software from the analysis.

In addition, due to the limitations of financial and time, we only managed to analyse the software that comes for free. The software tools that have been selected for the analysis are JGUIDev, wxwindows, visaj, Qt Designer Trolltech and lxb. Although the Qt Designer Trolltech and visaj can be considered as commercial software, the 30 days evaluation period sometimes constrain the analysis. For each builder, we will see their good criteria and what each is lacking from. All software are tested by designing some dialogues application.

3.1 JGUIDev 1.0, JGUIDev 1.1

JGUIDev, which stands for ‘Java Graphical User Interface Development’, is a software written in pure Java 2, that generates graphical interfaces in the form of easily reusable Java 2 classes. A huge advantage of this software is, it is a multi platform, which means it could run on any operating systems and machines. The JGUIDev 1.1 is the new version of JGUIDev 1.0, which it extends some functionality that already appears in version 1.0, namely the cut, copy and paste functions.

JGUIDev allows the user to create screens or dialogue boxes using the ‘drag and drop’ technique, i.e., the user choose the element from the toolbar, click on a mouse button, drag the item to where the user want it on the screen, and then release the button to ‘drop’

the element. The following diagram (Figure 3.1) shows what the interface looks like (version 1.1).

When JGUIDev starts, it displays a single main window that acts as the working space, which it is further consists of another two windows: Toolbar and Properties of a Component. JGUIDev has a basic mode of design, which is where the user builds the interface.

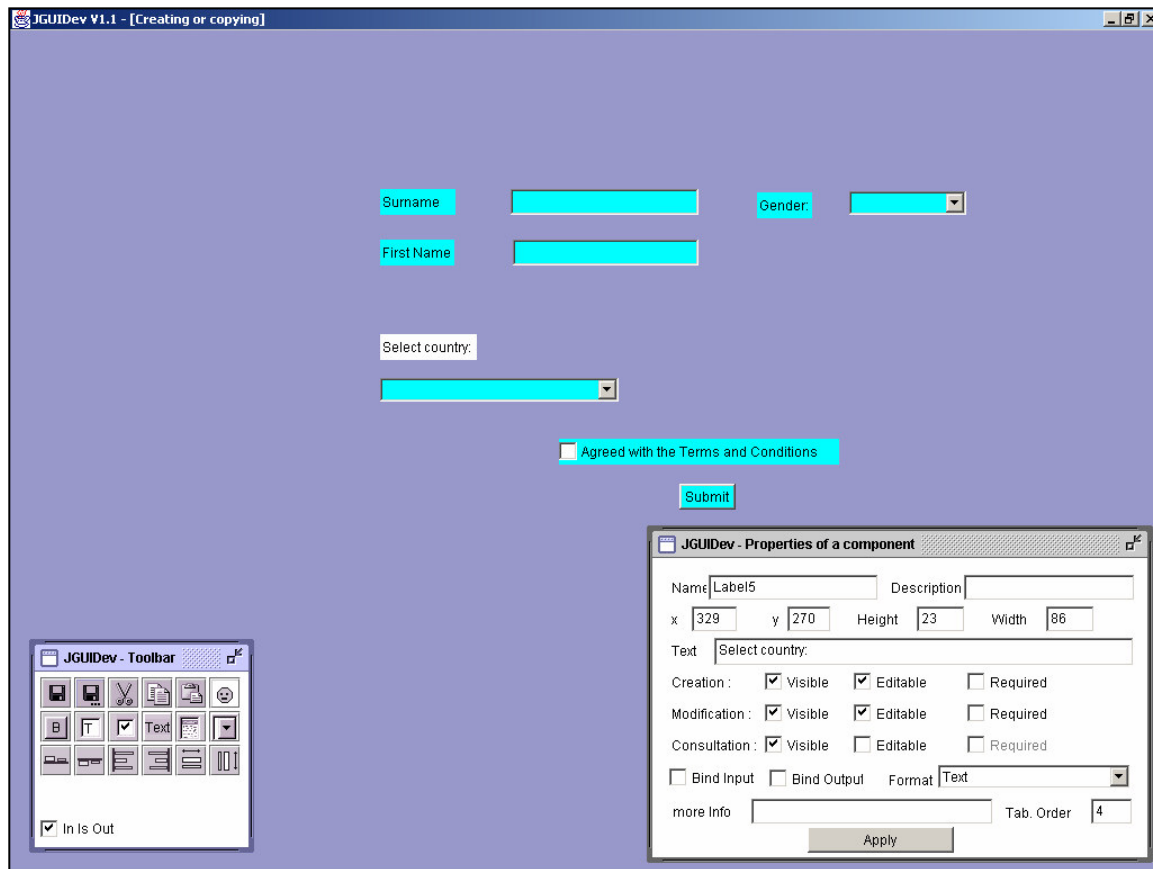


Figure 3.1: JGUIDev Screen Interface

The following are the results of the utilisation of both versions of JGUIDev:

- Although this builder claimed to have the 'drag and drop' technique, the fact is the movement of the components on the working surface is difficult, as each component keep flickering and does not stay rigidly on where the user wants it to be, unless the user give a click on the component before he or she could move on and do something

else

- The documentation or manual is not provided. Thus, it is very difficult to begin an interface. There is no information on how to use the 'editable', 'required' and 'consultation' options, which do not explain how these could be used by the user
- The work done could not be saved, and the given error message is: *Error while building attempt2:CreateProcess: javac attempt2.java error=2*. The error message is not understandable. The diagram overleaf illustrates this (Figure 3.2).

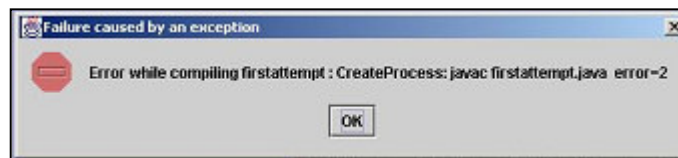


Figure 3.2: JGUIDev Error Message

- It has a very clean surface for the user to work with, and the colour scheme chosen is very appealing
- The minimisation of the window size, affects the appearances of the other windows reside in the main window. It would be more efficient if the toolbar and the properties windows were situated outside the working area

3.2 Qt Designer Trolltech

Qt Designer is one of the tools provided by a company called Trolltech Inc. All of its products, which are called Qt, are using C++ toolkit for application development. The chosen tool, Qt Designer, is a tool for designing and implementing user interfaces built with the Qt multi-platform application development framework.

Qt Designer is completely graphically oriented, so the user will be able to create directly windows, buttons, menus, etc. the user needs to write the codes that process system messages and events. Diagram overleaf illustrates the interface (Figure 3.3).

Qt Designer starts by displaying a single main window. It further consists of properties and object hierarchy windows. Both, however, can be closed, totally opt by the user.

The user begins an interface by selecting a new file, which give the user a number of selections to begin with, e.g. form, and upon the selection, the screen is displayed in the window. The icons or components are available at the bar of the main window.

The property Editor window enables the user to edit the values of the properties; the Object Hierarchy shows 'parent-child' tree corresponds to the interface.

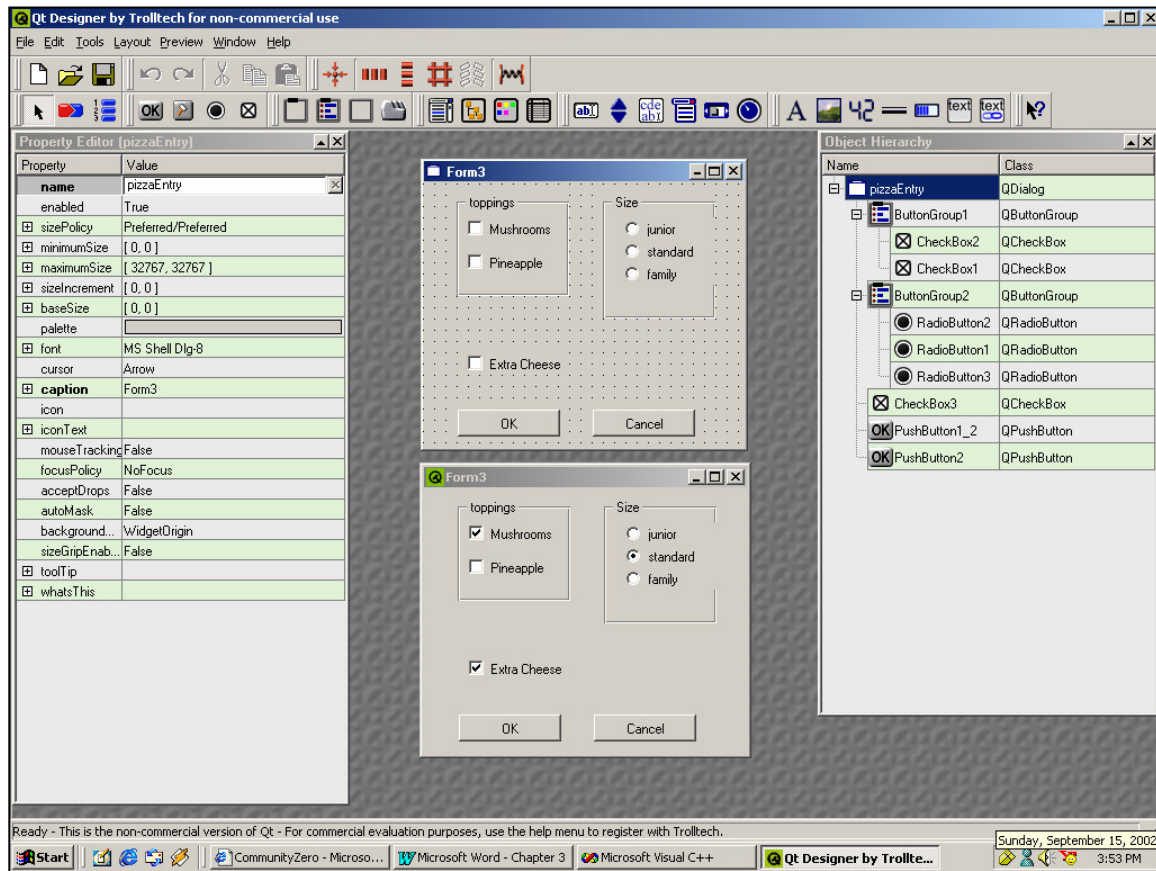


Figure 3.3: Qt Designer Screen Interface

The Qt Designer has two basic modes: design and test. The design mode is where the user builds the interface. Test mode lets the user feels the application of the design. It can be tested further, by including codes written in Visual C++.

The following are the analysis of the Qt Designer:

-
- It is very easy to use, as the manual or documentation provided is very clear in supporting the understanding of users. This is very much of help to the beginner users
 - The menu bars and components, which are positioned from left to right, are very much similar to the Microsoft Office software products. Therefore, in a glance, the user can start working with the tool without having to ‘think’ how to use it
 - The graphical icons and buttons are easy to handle and move. The smoothness of the movement produce more efficient work, which gives a higher accuracy in positioning the icons at where the user wants it on the screen
 - The components can easily be chosen by selecting the icons provided at the top of the screen, which it provides shortcuts to the type of buttons and icons by denoting with its picture on the top of the screen
 - The flexibility of this tool is that it is able to extend its functionality of the created interface with Visual C++ in Windows
 - It provides different windows, which float on the workspace that assist the users with the properties of the chosen widgets

3.3 wxWindows Dialog Editor

This tool offers a minimal amount of functionality to the users in creating a GUI application. When wxWindows Dialog Editor starts, it displays a single window that is divided into sections for dialogue hierarchy, which is situated at the left pane, and the selection of components at the bottom pane of the window.

The interface is begun by opening a new dialogue. The user then selects a widget from dialogue. The user can easily drag and drop, to where the user wants it on the screen. The widgets selected can also be grouped together in order to align them left, right or centre. The diagram overleaf (Figure 3.4) illustrates the interface.

The following are the analysis corresponds to the usage of the respective software:

- The top layer provides the user with the standard menu bars and options like, file, edit and help
 - The layer below the menu bar offers a significant number of functionality, which each
-

icon denotes a particular action, e.g. align components left, align components right, distribute components vertically and distribute components horizontally

- Widgets to create the interface dialogues are available at the bottom of the screen. Each icon depicts the action that each represents. This very much help the user in doing the selection
- The 'test dialog' option, that could be found from the Edit menu, gives the user the end results of the design dialog, which the user could 'feel' how the dialogue works
- One of the drawback is, the window size of the dialog is not resizable
- The tool doesn't allow multiple windows on the screen at one time
- The hierarchy situated at the left side of the screen, very much indeed, assist the user in giving them to keep track of what have been created. However, the icons listed in the hierarchy does not correspond interactively to the icons in the dialog diagram
- It provides a help manual which is very easy useful for the users to use before starting using the software

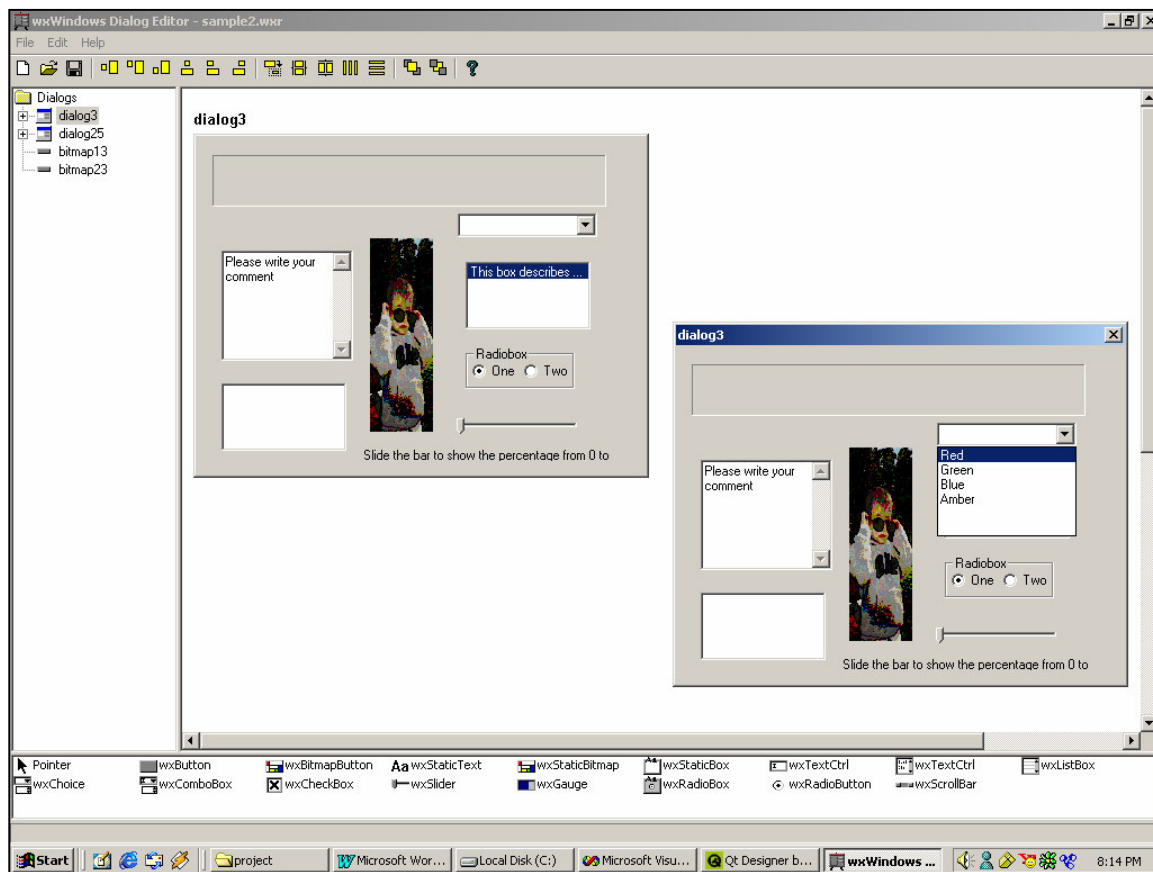


Figure 3.4: wxWindows Dialog Editor

3.4 Visaj

Unlike those tools that have been described earlier, this particular tool is more powerful as it allows graphical development of the structure and interface of an application. It helps the users in the sense that they do not need to write a single line of code. When the graphical representation is complete, the developer can generate Java code to incorporate the process message and events.

When Visaj starts, it displays a window that is divided into sections for interface (working area), palettes, and hierarchy. All of these are under the Beans tab. The other two tabs available are Event Bindings and Signature.

Under the Beans tab, the user could begin an interface by selecting a container from the toolbar, e.g. frame. The interface displays a tree hierarchy, which corresponds to the user selection that shows the relationship of 'parent-and-child'.

The user could be able to view design by clicking the 'binocular' icon situated at the menu bar. The properties are editable by clicking twice on the chosen components. The following diagram (Figure 3.5) illustrates the interface.

The Event Binding tab lets the user to create an event for particular component. Therefore, the user does not need to write any single code.

The list below describes how the design of this software affects the usage by the user:

- This tool is very easy to use, once, the user has gone through the tutorial provided from the 'help' menu
 - The layout is very organised, which the selection of widgets are provided on the left side of the screen
 - The hierarchy very much gives the user the sense of knowing what is happening throughout the design process
 - Again, the 'left-to-right' scheme has been applied
-

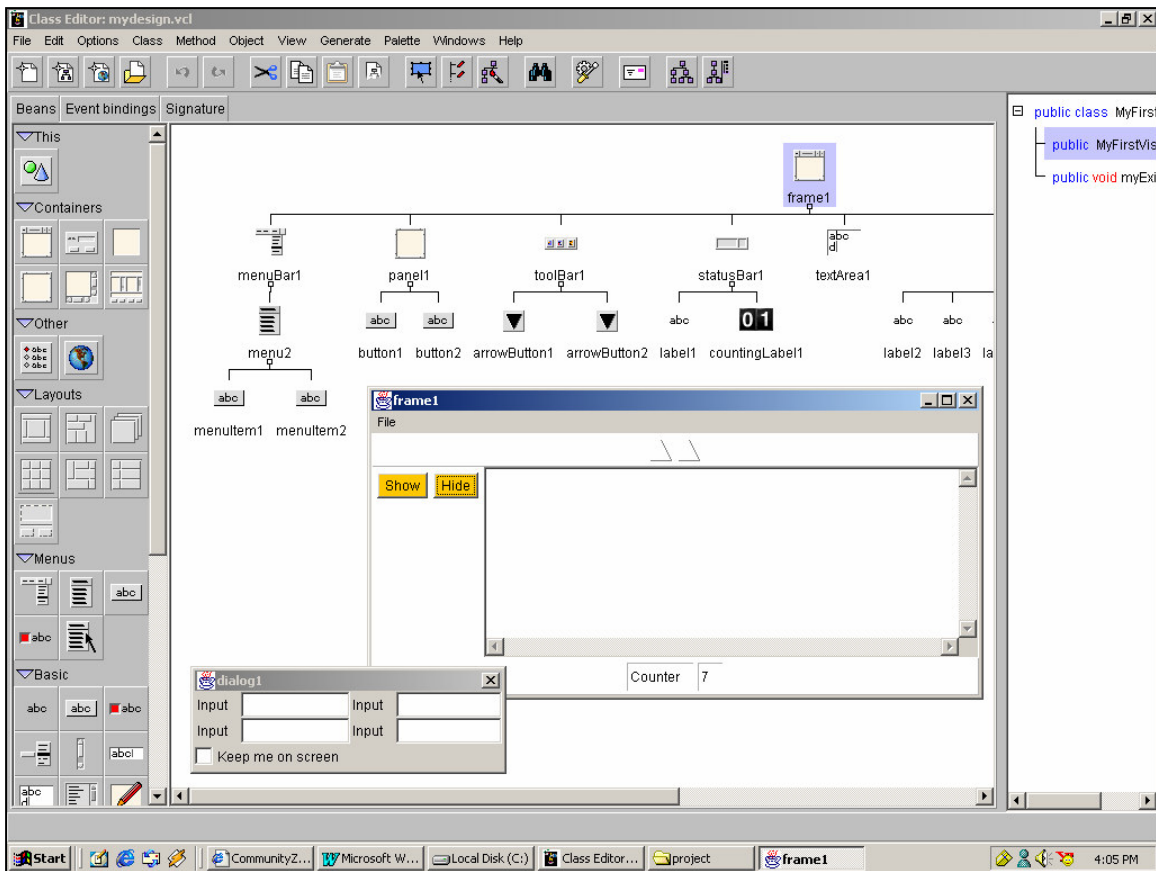


Figure 3.5: visaj Screen Interface

3.5 lxb

This software is UNIX based platform. It consists of three windows; Application, Widget Palette and Resource Editor. The Application window is the first window the user needs to look at, as it provides the file creation, and saving. Here, it also shows a hierarchy of the design that is being developed. In order to build a widget, the user first needs to select a frame from the Widget window, as every GUI component must be contained in its own Frame, and there can only be one GUI component in each Frame. Resource Editor lets the users to amend the GUI component's name, and values. Overleaf illustrates the lxb interface (Figure 3.6).

The list below found from the usage analysis of lxb:

- It is easy to use, once, the user has gone through the step-by-step guide, in order to

understand the concept of how the tool should work

- The user can move and drag the created widgets on the working area and position them on the desired place
- Once the design is done, the user could test and 'feel' how it works by choosing 'Play Mode' from the main menu.
- It doesn't provide Help or Manual documentation
- The Widget Palette does not give the user much insights of how the GUI component might be looked like. Graphic image of each widgets would have helped

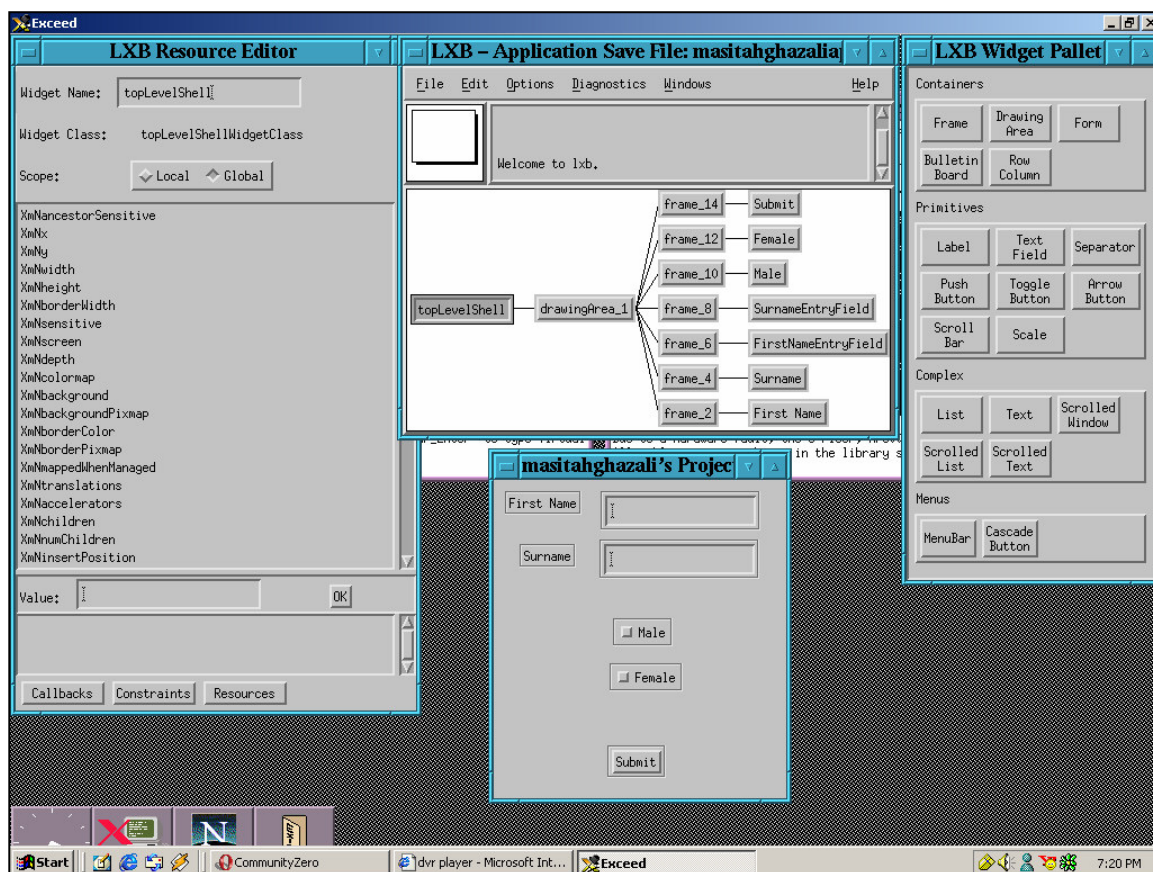


Figure 3.6: lxb Screen Interface

3.3 Conclusion

Having thoroughly analysed the different kinds of GUI builders have enabled us to capture and attain much insights, greater ideas, and information about GUI builders. In addition we are able to discover each software builder's strong features and weak points.

Furthermore, this chapter fulfilled its objectives to discover the criteria that make a good GUI builder. These findings will be used to produce the generic guidelines, which will be elaborated further in the Chapter 5.

Chapter 4: Survey on lxb Software

This chapter focuses on the analysis of the survey operation, which has been carried out on 70 students. The participants involved with this survey were of the second year Computer Science students from the session 2001/2002. These students were required to test and evaluate the lxb software by undergoing two sets of questionnaires that are web-based oriented. These user interface evaluations with questionnaires has been set up by Gary Perlman¹. This particular analysis is essential and is one of the criteria of this project in order to attain the results, which leads to producing the guidelines. The aims of this survey are twofold. First is to obtain the usefulness of the lxb software. Second is to inspect to what extent the GUI really constrains the design, for better or worse.

4.1 Web-Based User Interface Evaluation with Questionnaires [11]

These questions are a customisable web-based Perl CGI script to administrator and collect data according to a few "standard" User Interface evaluation questionnaire forms, e.g. Questionnaire for User Interface Satisfaction (QUIS), Perceived Usefulness and Ease of Use (PUEU), Computer System Usability Questionnaire (CSUQ) and Purdue Usability Testing Questionnaire (PUTQ). The options used by the scripts include:

- (i) "general customisations" such as system name, administrator email,
- (ii) "customisations to the rating scale" such as number of points, labels, and
- (iii) "number of open-ended positive/negative comments requested, to present the questionnaire on the Web, gather ratings, general comments and question-specific comments, and email the results to the designated survey administrator".

There were two sets of questionnaire used to evaluate lxb. These are the Questionnaire for User Interface Satisfaction (QUIS) and Computer System Usability Questionnaire (CSUQ). The QUIS was designed to assess users' subjective satisfaction with specific aspects of the Human-Computer Interface [1]. Both concerned with the empirical values to evaluate a system. More about QUIS can be found at [1] and [12]. The way the CSUQ

¹ He is a senior consulting research scientist at OCLC [10]

differs from QUIS is, CSUQ “measures the users’ satisfaction with the usability of computer systems in the context of scenario-based usability studies” [2]. Further information on CSUQ is accessible at [2].

4.2 QUIS Results²

This latest version of QUIS (version 7.0) has divided the questions into several sections. The scale varies from 0 to 9, which anchored at left end with negative criteria and at the right end with positive criteria. There is also a NA option for items that the students may think are not applicable to the software. The questionnaire is accessible from [12].

4.2.1 Overall Reaction to the Software

Almost the entire student felt the software is terrible, as the software is not so user-friendly. Despite this fact, they found lxb to be not too difficult to use. It is also not too easy to use due to the fact of the insufficient amount of documentation.

The students also felt disappointed, as even when the system is once learnt, it is just simply frustrating to use. It is even more frustrating whenever the software crashes, which this usually followed by losing the work they have done.

It is presumed that those reasons influenced the students' opinion to say that lxb software to be inadequate in power. Regardless of this presumption, lxb is still recognised to be able to produce any dialogues than could be imagined by the users.

'Dull' is the adjective the students had preferred to choose stimulating, when describing the overall reaction about the system. The most obvious reason is, there was no single graphics provided (this is before designing take place), especially in attempting the display of widgets.

The survey shows that the software is not flexible enough, as it has no ability to link between the dialogue boxes.

4.2.2 Screen

The consistency of reading the characters on the screen has been observed to be not so consistent. It is felt that the fonts used are too small and not of the right type. Therefore, the students had some difficulties when reading them from the screen.

'Highlighting simplifies task' is one of the important criteria. The Resource Editor applies this criterion to show the users, which frame or field they have entered the values to in the Editor. Without this, there is no indication to the students which particular value has been selected. Nonetheless, sometimes, when highlighting is taking place, object text disappeared and often makes items illegible that this complicates the completion of work.

Organisation of information and sequence of screens are both neither confusing nor clear. The sequence of screen is quite confusing. The reason being is the three screens need to be re-organised every time the application starts, plus, there is no logical sequencing between them.

4.2.3 Terminology and System Information

The use of terms throughout the system is consistent, despite the fact of the existing of some ambiguous terms found in the widget attributes list and within the error messages. The terminology related to task and the position of messages on screen, show moderate characteristics respectively. One example that points out the inconsistency of messages positioning is, they sometimes appear in the console and sometimes as pop-us.

Prompts for input is balanced between the confusing and clear criteria. It is quite confusing, as the users need to change the default values each time a new widget is added due to the poor conditions. This, however, is balanced with prompts for input in the Resource Editor. The Resource Editor is clear in the essential feature required to develop a user interface.

² Refer to Appendix A and B for the full results

The computer never informs about its progress to the users. All it does is showing the tree hierarchy, which tells what is happening in console.

In the meantime, the error messages were unhelpful, rather than being helpful. The messages were incomprehensible, and only appeared after the system had crashed. This does not help much as it is impossible to correct the mistakes. In addition, the messages suggest no way for solutions.

4.2.4 Learning

The students felt it was easy in learning to operate the system and is quite easy for them to remember names and used of commands. The consistency of using terms throughout helped much in these subjects.

The users perceived the exploration of new features by trial and error as moderate, i.e. it was not too difficult and not too easy. The simplicity characteristic of the system helps a lot to that direction, except when it crashes.

The poor performance of performing tasks, help messages and reference materials are those weakening the learning process of lxb software, according to the survey. Performing tasks are almost never straightforward, and the help messages on the screen were unhelpful to the students.

The supplemental reference materials are thought to be confusing, rather than to be informative. This fact, however, perceived to be not applicable to be answered related to lxb by quite a significant number of students.

4.2.5 System Capabilities

The system capabilities as a whole are poor. Firstly, the system is too slow. The performance is quite terrible, as the students had to regularly wait after clicking on certain changes.

Secondly, the system is unreliable as it often crashes, and once it crashed, there was no recovery of the work. Thirdly, it does not provide any facilities to correct the mistakes done by the students.

The design of the system does not suit all levels of users. It is presumed that those who are not from computing background would face some difficulties to use the system.

To give opinion whether the system tends to be noisy or quiet, almost half of the students chose NA (not applicable) item. The description of this criterion is not that clear to be answered by the students.

4.3 CSUQ Results

This section describes the students' performances towards the implementation of lxb, in which how they respond to each item by scoring the appropriate number. The results were then converted into percentage³. The questionnaire is available at [3]. Each of the questions had rating scales ascending from 0 on the left to 7 on the right and anchored at both end points with strongly disagreed and strongly agreed.

The analysis results of the QUIS below have been categorised into seven different categories. These groupings are based on their similar characteristics. The identified categories are easiness, completion of work, users' satisfactory, error, information, interface, and overall performance.

4.3.1 Easiness

Nearly over half of the students⁴, were not satisfied with the difficulties they were facing when using the system. Meanwhile, some positive answers were obtained when asking about the simplicity and the learning process of learning the system. The learning process however, has a steep learning curve, but with perseverance, it is easy to use. Among the

³ Refer to Appendix D and E for the full results

⁴ 54.0% of the students opted score 1, 2, and 3, which these lean towards the 'dissatisfied' stance

concerns that have been raised are, lxb does not have the undo function, which makes it harder to use, and the easiness of learning how to use the system may be found a bit difficult for those who are not of computer scientists or computing students.

4.3.2 Completion of work

Of all the students, 69.8% of them did not find the system enable them to complete their work effectively. This is due to the fact of the existing bugs, which disallowed the completion of certain tasks. Nevertheless, even when the system was able to complete the work, the results were not that well.

The students also did not able to complete the work quickly and efficiently when using the system. The system is reckoned to be slow, even when doing the most mundane tasks. Owing to this fact, it is therefore very time consuming. The lxb's unreliability, rigidity and slow reaction-speed of system, also reduced the efficiency of the task completion.

4.3.3 Users' satisfactory

Majority numbers of students do not feel comfortable using the system, and they also believed that they did not become productive quickly using the system. The uneasiness feeling is caused by the un-stability of the system, which this leads to losing the users' work. The users only became productive once the basics were learnt, and not until the system began to crash and lost the work.

4.3.4 Error

It has been clearly shown from the results that the given error messages do not offer any resolutions to problems nor any explanation to the students on how to fix the problems. Furthermore, the students found that whenever they make a mistake using the system, they did not recover quickly and easily. Yet, lxb normally results in a crash.

4.3.5 Information

From the survey, the information provided, such as the online-help, on-screen messages and other documentation, were not helpful, let alone to be clear. In addition, the 'help

menu' was not available, i.e. empty. The insufficient amount of information sometimes gave difficult time to the students to work around the software.

Even though there is some information provided by the software, most of the students find it difficult to find the information they needed from the system. In this particular case, the students could only rely on the information provided by the lecturer, i.e. a tutorial, but not from elsewhere, e.g. web sites.

When it comes to answering whether the information provided for the system is easy to understand, or not, and whether the information is effective in helping the students complete the tasks and scenarios, the students shows moderate answers. They did not find them easy as most words are in jargon, or not present at all, but these are compensated by the information provided by the lecturer.

The students also gave a moderate view when answering whether the organisation of information on the system screens is clear. The students complained about the ambiguous windows, which are often offset (the three screens overlapped one another when the software is first started), requiring the dragging back into the centre of the screen.

4.3.6 Interface

Most of the students find the interface of this system pleasant. Nonetheless, the students pointed out some weak points about lxb software that are to worth to be considered in order to improve the system. They are; there were too many windows to organise, there was a lack of icons and some graphical presentation was confusing. Although the students find the interface as pleasant, they don't really like using the interface of this system as it crashes easily.

4.3.7 lxb as a whole

Almost all of the students think that this system does not have all the functions and capabilities they expected it to have. They raised the issues of the system cannot link the dialog boxes, short of undo function and lack of help information. All in all, most of the

students did not satisfy with this system due to the fact that it is slow and very time consuming.

4.4 Negative and Positive Aspects of lxb resulted from QUIS and CSUQ⁵

Both aspects; negative and positive, have been further categorised into sub-categories, e.g. performance of the system and interface, in order to recognise the requirements that the system still lack of (negative aspect) and the features that the system already has as its powerful characteristics (positive aspect).

4.4.1 Negative Aspects

The negative aspects below have been divided into five different groups. They are; the performance of lxb system, its features (functionality), layout and design, the provision of online help or documentation, and feedback. Also included are, some comments received by the students.

Performance

The lxb performance as a whole cannot be asserted as remarkably excellent. As we have previously seen, the word 'crash' seemed to appear quite frequently in the survey. The instability state of lxb is very much affecting the work flow of the students. The citations below received by the students, which respond to this aspect.

"... This system is unstable, i.e. it crashes regularly for no apparent reason and will not reopen file..."

"... It is time consuming, which it is too slow when making changes..."

Apart from the above comments, the performance is also regarded to have flaky performance, inconsistent behaviour, inflexible, and has unreliable storage of files.

Features

The most noticeable disadvantage this system has is the lack of editing facilities. It is lacking of the undo or revert functionality and cut-and-paste. These features have been

proved to be amongst the most important aspects in all generators or any other applications, which it facilitates the speed of the work. The lacking of these features and the fact of others functionality disabilities to support what they offer, coupled with its un-stability condition, have made the overall functionality even worse.

Below are some complaints obtained from the survey regarding the undo and other functionality:

"... I can't undo or revert errors which this loses my work..."

"... The application does not appear to be able to support the functionality it offers, such as implementing drop down menus when switched to 'play mode' causes program to crash or makes files un-openable if saved..."

Layout / Design

When the system is first started, the three windows or screens (Main, Palette and Attributes) were overlapped one another. This normally confused the first time users as they find them a bit tricky at the beginning. They have to allow some time to get around the software in order to get familiar with its layout before they could start with their work. Below comments support this fact.

"... The interface is a little tricky to learn at first..."

"...The initial layout of screens is confusing and small, which I need to readjust them everytime the lxb starts..."

Other drawbacks related to this aspect are; the poor design of the interface, the difficulties in understanding the text, widget attributes and options. Having mentioned in the previous section, the text is quite small to be readable, and the widget attributes are filled with jargon names.

Online help / Documentation

As been described earlier, the students were only able to learn how to use the software by referring to the tutorial provided by the lecturer from their lecture. Apart from that, there

⁵ Refer to Appendix G for full results

is no provision of online help or any other documentation in assisting the users by the software. Although help systems and documentation are normally perceived to be two different meanings⁶, we must admit that both do guide the users how to do their job. Admitting this fact, this system does have 'Help' in the menu bar, but, ironically, it actually contains nothing.

The citations below support this fact:

"... There is no help feature for querying different features and options..."

"... There is no clear explanation on how to use the system..."

Feedback

The existence of feedback in one system should be helpful and meaningful. Regardless to this fact, the existing feedback in lxb software appears to be 'useless'. Instead of notifying the errors that have been made and solutions, the feedback only appear to indicate an error has occurred. The comments below show the disappointment feelings of the students:

"... The error message was unhelpful with no chance to fix errors..."

"... It does not prompt the user when there is an error..."

4.4.2 Positive Aspects

There are six sub-categories that have been identified to be existing in lxb software. Four of them are of the same groups as before, which are; the features and its functionality, its layout and interface, feedback and performance. The remaining sub-categories are the learning aspect and its concept.

Learning

Despite the fact of facing some difficulties when first using the system, the students did admit that the system is relatively easy to learn due to its simplicity and the provision of the tutorial by the lecturer. By closely adhering to the guidelines, the users were able to understand and get the general concept of the software and after a short while they could independently do some basic tasks. The citations below support this:

⁶ "Help systems are problem oriented and specific, whereas documentation is system oriented and generic"

"... lxb is fairly intuitive and relatively easy to learn ..."

"... the learning time required is short ..."

The students also noted that this system used quite simple action verbs throughout and simple descriptions. Thus, making the learning process more comfortable.

Features / Functionality

The Palette window offers a selection of widgets to the users, which these are beneficial when designing the dialogue boxes. Furthermore, the Resource Editor enables the users to change the attributes of the selected widgets, such as the width, length, value, title, etc. The satisfaction is clearly flaunted from the comments below:

"... The program appears to have a lot of features allowing for advanced users to meet their initial designs quite easily while maintaining ease-of-use for new users..."

"... It is able to change the colour..."

"... It has lots of options and widgets available..."

"... The Resource Editor is a good way of editing an item properties..."

Layout / Interface

In the main screen of lxb software, contains a hierarchy, which depicts the design process. In other words, it displays the relationship between one widget or component to another. This keeps the users on track on what they are doing as the hierarchy is so easy to apprehend and logical. One of the student said, "...it is logical, which the program has a very logical flowchart layout..."

The way the three screens distinguished themselves from one another has made the organisation of the information very clear to the users. Thus, this enable the students to easily comprehend of what each screen does, as this comment claims; "...the organisation of information is quite good..."

Feedback

A good aspect of the feedback from the user point of view is the system does provide some kind of indication when something has happened. The rapid feedback notifies the users of the changes. Following is one of the comments that support this fact.

"... It is easy to see result of changes..."

Performance

Omitting the fact that the system often crashes, the speed of the system is actually quite fast, as one of the students cited: "... when system is running fine, the output is quite fast...". This fact however, only occurred once in a while, as it usually crashes rather than running fine.

Concept

This sub-category is basically concerned with its non-functionality or the characteristics of lxb software. As the participants of the survey are of the computing students, they have identified the application of the direct-manipulation in the lxb software. The applied direct-manipulation in this system is thought to be reasonable for the users to utilise, as they said. "... Some reasonable direct manipulation principles applied here like selecting widgets from the palette..." and "... I have the ability to manipulate objects..."

The simplicity concept is consistently used throughout the system. This gives much ease to the students when designing the dialogue boxes. The citations below prove this.

"... The concepts within the program are used consistently, which the elements within frames and other concepts are not difficult to grasp and utilise..."

"... Better than programming each interface item manually..."

"... Separation of like tasks (different tasks are easy to carry out)..."

4.5 Conclusion

The results of the survey are significant as the two questionnaire sets cover every aspect that one needs to know about a system, and they are even more significant as the results were obtained from the users' perspective. It also succeeds in meeting the two aims that

have been mentioned earlier in the introduction, to see the software usefulness and to what extent it constrains the design.

The survey's results unleashed the usefulness and drawbacks of lxb. The most prominent usefulness that lxb has is its 'flow chart', which enables the users to see the relationship between one component to another.

From this chapter, we could make two assertions. First, the lxb software affect the design when they are no longer serve the purposes it supposed to serve, unable to fulfil the users' needs, or misleading from what it should behave. Secondly, as lxb is one example of interactive systems, it constrains the design by imposing the design to be always applying the concept of direct manipulation.

Chapter 5: Generic Guidelines

This chapter would be the essence of this project as it highlights the generic guidelines for the GUI builders. The first part of this chapter delineates the steps taken to derive to the guidelines. The next section briefly describes all the potential criteria and factors that are important in producing the guidelines, by relinking or relating them back to the previous chapters: the Literature Review, Analysis and Survey. The final section lists the generic guidelines of GUI builders.

5.1 Derivation Steps

Basically, there are two stages that had to be taken before the generic guidelines are derived. Firstly, all findings from chapters 2, 3 and 4 were re-collected. This is then was followed by removing any redundancies of criteria found from the 're-collection' procedure. Figure 5.1 below illustrates the steps taken to produce the guidelines.

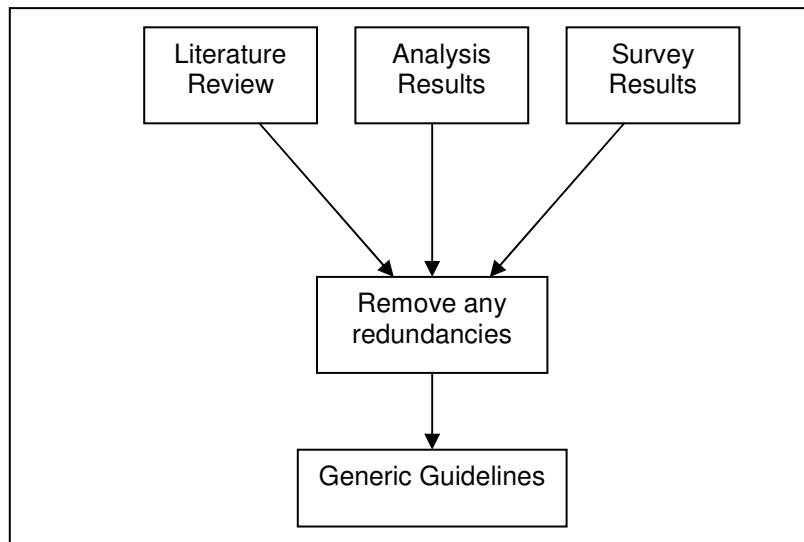


Figure 5.1: Steps taken to produce Generic Guidelines

5.2 Brief Descriptions on Criteria

Having analysed the Literature Review, Analysis and Survey chapters, have given much insights of the good criteria of GUI builders. This section points out all of the criteria, which comes with a brief description of each.

5.2.1 Multiple Windows

Multiple windows on the same screen would give the users choices whilst working on the design. The windows could be of the toolbar, properties or hierarchy windows, besides the working area window itself. The existence of multiple windows with each represent different tasks would enable the users to identify and distinguish the tasks easily. The recommended amount of number is from 3 to 7.

It is also important to ensure that the windows have the ability to minimise and maximise, and close, which this is supported from we have seen in section 2.3.2 earlier. The windows must also be dependant to the main screen, for instance, once the main screen of the GUI builder sets to minimise, the rest of the windows that reside in it, must also be automatically minimised.

Multiple windows is also important in order to support different types of tasks. A good example of this is Qt Designer Trolltech (section 3.2) and lxb software (section 3.5) due to the reason that they provide a large working area, which further consists of properties and hierarchy windows. In addition, these windows are optional to the users, which they could actually turn them on or completely close them up. JGUIDev software does not allow the toolbar and properties windows to be minimised when the main screen is set to smaller size (see section 3.1).

The Visaj software doesn't have multiple windows (see Figure 3.5 in Chapter 3). However, it compensates this by offering the different tasks in different panes. For instance, it offers the available widgets on the left pane and the hierarchy trees on the right pane, which the working area situated in the middle, separating these two.

Some difficulties can be experienced by the users when dealing with multiple windows if the concept of multiple windows is misled. We can trace back the examples of overlapping windows from sections 4.4.1 and 4.4.2 of the layout/design columns, where the three screen windows of lxb software confused the students.

5.2.2 Consistency

Section 2.3.1 has emphasised the importance of striving on consistency. The consistency may encompass the principles like colour usage, and the positioning of the options tabs and menus. Colour usage, as what has been described in colour column in section 2.3.2, may sometimes thought to be the least of the problem contributes to the generator judgement. This is untrue, as the right selection of colours is able to make the users to stay longer to use the system. Imagine when the GUI builder uses bright orange or bright green as its theme colour. This easily put the users' enthusiasm off to continue to use the system. Of the most appealing colours used nowadays are white, beige and those that are of pastel colours. This, of course, does not include the colours of the icons or buttons which some colours represents their own meaning, e.g. red denotes alert or aware.

The analysed GUI builders used different kind of colour scheme. JGUIDev uses lilac-ish colour, which this is very interesting to keep the users motivated in using the software, regardless the fact of its performance (see Figure 3.1). Colours like light grey and white are amongst the common colour used in many GUI builders and this usage has very much proven their credibility in appealing their users.

Of all the GUI builders, Qt Designer Trolltech (see Figure 3.3) was the one that has the most significant colour scheme. It comprises different colours. The background of the working area is of a pattern in grey-ish colour. The windows on the left and right of the screen are of light green colour and the dialogue box is in grey colour. All of these signify the different zones or areas of working.

As most of the software available today consists of menus and toolbars, the same positioning should be applied in the generic builder. The renowned 'left-to-right' positioning enable to keep the users' pace fast and to avoid any misleads actions.

The consistency is also important in the application of reading the characters on the screen, like the one that has been stressed in section 4.2.2. The consistency in using terms throughout is also vital in terminology and systems information, as to keep the users

familiar and understand the system faster and easier (see section 4.2.3). It is also essential in process of learning (see section 4.2.4).

Each of the analysed GUI builders implies the 'left-to-right' positioning layout of menus, except the JGUIDev software tool (see chapter 3). This type of positioning is felt to be vital because of its potential to let the users understand the way around the software. Section 2.3.2 on standardisation, also shows the same concerns.

The positioning of the available widgets could also be of benefits to the users when they were positioned just underneath the standard menus, like in Qt Designer Trolltech software in section 3.2. Familiarity concept is essential in order to ensure the users do not need to spend too much time to learn about the new builders.

5.2.3 Meaningful Feedback

Short sentences, yet meaningful is very much efficient in order to ensure the users understand what is actually happening. Section 2.3.1 and 2.3.2 have stressed in the importance of having meaningful feedback in a system. If the feedback is of the error message, it should avoid from giving a technical feedback message or code error message. It would be of much assistance, if the feedback also pinpoints the exact location, which raises the fault. The feedback should also be rapid, as section 2.3.2-rapid response, has emphasised its importance.

Less meaningful feedback would lead the users to nowhere. Section 3.1, which has the example of this scenario, is described as follows. When the procedure of saving a work from JGUIDev fails, an error message appeared. It is a good remark that it provides the user with this message; however, the message is not understandable by the non-Java programmers. Instead of quoting the error that refers to the line of code, it should tell the user what type or kind of error that user has done. Therefore, the user could give it another attempt according to the method.

It would only frustrate the users if the system fails to provide meaningful feedback. Section 4.2.3 and 4.2.4 describe the example of unhelpful error and help messages that existing in lxb software. Further comments from the users regarding this are also available in section 4.3.4, 4.3.5, and 4.4.1.

5.2.4 Online Help

Online documentation or manual is very important in software. Normally, the novice users rely on this option before they could begin with their work. This online help might not just include the tutorial, but it also should have included an index of the contents, i.e. the help context-sensitive, that elaborates further the available options. We can assert that it is a must to provide 'Help' in the menu.

The provision of 'Tutorial' under 'Help' menu will be an advantage, as it enables the novice users to begin an interface. Comparing the JGUIDev software tool with Visaj generator proves this fact. JGUIDev software, which does not provide any of these, has led the users in difficulties when using the software (see section 3.1 and 3.4).

The supplemental reference materials also need to be crystal clear, helpful, and sufficient, or otherwise it will confuse the users. Section 4.2.4 shows the difficulties in which the users experiencing when the materials are not informative. Other examples are available from section 4.3.5 and 4.4.1 (online help/documentation).

5.2.5 Target the User

As the users would be of different background like, developers, programmers, and end-users, the design of the GUI builders should be minimal and simple in general, in order to involve every one of each group. If some processes seem to be difficult, the online help should be able to guide them.

Some difficulties would be faced by the users if the design does not cater for all levels of users. This is what the users have claimed when evaluating the lxb software (see sections 4.2.5 and 4.3.1).

5.2.6 Accuracy

The smoothness movement of dragging the widgets is very essential, as it results to positioning to each component at the actual position. This criterion correlated to the drag and drop technique describes in section 5.2.7. The accuracy plays such an integral part due to the fact that the designs of the developed dialogues in the working environment mimic the actual design of the dialogue boxes. Example of the application of accuracy concept can be seen in section 3.2.

5.2.7 Cut and copy, drag and drop techniques

Most of the analysed GUI builders apply this technique (although JGUIDev is able to perform this technique, the performance is very poor - see section 3.1). It offers convenience to the users, as they could easily place the widget(s) to where they want it(them) to by simply choose, cut and paste, or, choose, drag and drop. Without the presence of this techniques, makes the working process much difficult and slows down the speed of work (refer to section 4.4.1).

5.2.8 Shortcuts

In order to save time, shortcuts would be one of the solutions. Sections 2.3.1 and 2.3.2 under column Menu, have pointed out its importance. Apart from providing the applicable widgets and components under the menu options, the available widgets could also be offered to the users in more realistic way. This means that the image of the widgets can be positioned next to the working area, so that the users could easily choose by simply look at the icon of the widgets.

The Qt Designer Trolltech (section 3.2), Visaj (section 3.4) and wx windows (section 3.3) software adopts this idea. The lxb (section 3.5) shows the components/widgets only by naming each on an icon. This is not sufficient, as there are no images depicting each of the components.

5.2.9 Preview

Having tested all software as described in chapter 3, it is felt that Preview is such an important criteria, in which each GUI builder should have this in its system. Preview enable the users to 'test' the dialogue boxes, which in this state, the users are now able to test and feel the designs they have developed earlier. Of all software that have been analysed, one that has not incorporated this criteria into its system is JGUIDev (see section 3.1).

5.2.10 Hierarchy

The existence of the hierarchy window in the main screen certainly assists the users in keeping track of what components or widget they have added into their design. Apart from that, by having this, the users could also see the 'parent-and-child' relationships between one widget/component to another. The software like, Visaj and lxb, apply this idea and display the hierarchy in the tree-list format (section 3.4 and 3.5).

5.3 Generic Guidelines

This section puts the findings together in order to produce the generic guidelines of the GUI builders, which this also permits us to see to what extent the GUI builders affects the design.

First, it is important to have an overview of the ideal concept of GUI builder. From the findings, the software must be able to, or must be:

- user-friendly
 - easy to learn, as well as to use
 - stable (does not crashes)
 - stimulating
 - flexible
 - fast (speed)
 - reliable
 - correcting mistakes
 - for all levels of users
-

-
- consistent, i.e. colours, terminology, layout
 - provide rapid, meaningful feedback

Therefore, some aspects must be taken into account in the layout of the interface design and the architecture⁷ of the software. Those are:

- minimal number of windows
- clearly distinguish windows according to their tasks, and give each of the windows a title
- moderate amount of text
- organisation of information
- flowchart or tree diagram
- image icons that depict the widgets
- grid

The functionality and significant features must be embedded into the GUI builders in order to complete the software. What the software must at least have are:

- to be able to link dialog boxes
- save the work safely, which comes with confirmation
- easy to move widgets
- easy to correct mistake
- able to cut and paste
- able to undo, or revert
- editing and if possible by clicking right click of the mouse
- able to create several screen within a file
- help documentation
- portability
- shortcuts

⁷ The architecture is primarily concerned with the direct manipulation concept. This concept will be further explained in the next chapter.

5.4 Conclusion

This chapter has clearly shown the derivation of the generic guidelines, which the findings were taken from the previous chapters: Literature Review, Analysis and Survey. In general, GUI builders have one solely purpose that is to be able to design and create system dialogues. Therefore, the generic guidelines have meticulously identified the rules by incorporating the important criteria, which the GUI builders must have in their system. The next chapter focuses on the design of a proposed GUI builder, which this follows the generic guidelines.

Chapter 6: Design

In this chapter, we will see how the findings from the previous chapter (Chapter 5: Generic Guidelines, section 5.3) are being integrated into the design process. The first section discusses the concept of direct manipulation, which will be applied in the design, and its importance in the GUI builders. The following section contains the elaboration of WIMP interface as the interaction style for the builders. A State Transition Diagram (STN) for designing dialogues will then be presented in the following section. The final section presents the proposed screen design of the generic GUI builder.

6.1 Direct Manipulation

In the previous chapters, we have seen the direct manipulation term is mentioned several times by the students from the survey (section 4.4.2 - concepts) and in the Literature Review (section 2.2.1). Now we will discuss why this concept is important in the application of GUI builders. The existing of direct manipulation concept in any interactive systems has been proved to be important as it offers rapid feedback. One of the example of rapid feedback is, when the user does something, he or she will immediately gets the results correspond to their action.

If we refer to Shneiderman's features of a direct manipulation interface [14], rapid feedback is only one of them. Below is the full list of the features:

- visibility of the objects of interest
- incremental action at the interface with rapid feedback on all actions
- reversibility of all actions, so that users are encouraged to explore without severe penalties
- syntactic correctness of all actions, so that every user actions is a legal operation
- replacement of complex command languages with actions to manipulate directly the visible objects (and, hence, the name direct manipulation)

By comparing the similarities of the generic guidelines from the previous chapter and the list of the features above, we know that the GUI builders' concept is of the direct

manipulation. The reason behind this is due to the 'Interactive' characteristic of the GUI builder itself. This interactive characteristic is closely related to the interaction of the least of two participants: the user and the system. Diagram below, which is adopted from [4] shows the interaction framework of the System (S), the User (U), the Input and the Output.

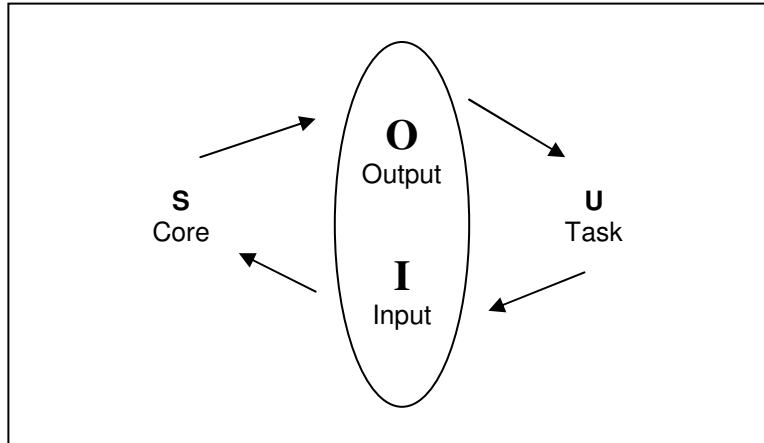


Figure 6.1: The general interaction framework

Therefore, the direct manipulation concept plays such an integral part in the development of the GUI builders.

6.2 WIMP Interface

Most of the available interactive systems today apply WIMP interface, which is also known as "windowing system". It is one of the most common interface styles in the development of interaction system⁷. WIMP actually stands for windows, icons, menus and pointers [4].

The WIMP interface will be applied in the design development of GUI builders. As we have noted earlier from the generic guidelines, the GUI builder indeed needs to be of windows, icons, menus and pointers in order to fulfil its own objective of being as GUI builder. As such, the GUI builder will also be of some other additional interaction objects like buttons, toolbars, palettes and dialog boxes.

The application of WIMP interface will be discussed further in Section 6.4.

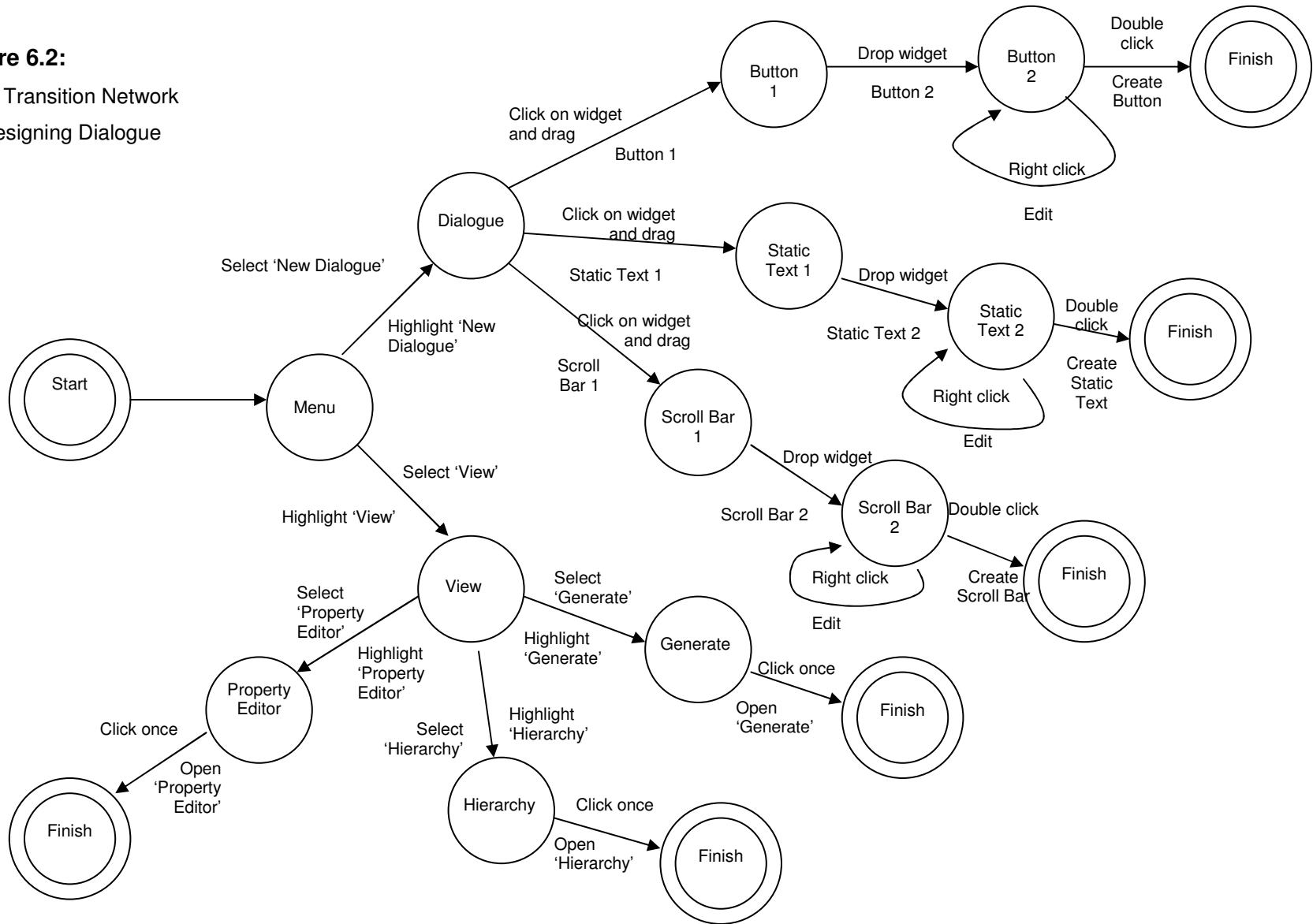
6.3 State Transition Network (STN)

Revisiting Figure 6.1 diagram, we noted that between the system and the user, there rely inputs and outputs of the orders and structures, which is also known as the syntactic level of dialogue. The role of State Transition Network (STN) in this stage is to represent the diagrammatic dialogue notations, which has the potential to “allow the designer to see at a glance the structure of the dialogue” [4].

Figure 6.2 overleaf illustrates the STN of the proposed design GUI builder for designing a dialogue. It consists of circles, which denote the ‘states’ and arrows, which denote the ‘transitions’. The STN below starts with the state Menu, which in this state, the system is waiting for the reply from the user, either to select the Dialogue state or View state. When the user selects Dialogue, it goes through what we called as transition, which from the figure below it is labelled with the user action and respond that the system makes.

In the Dialogue state, the user has the options of selecting various types of widgets (figure overleaf, however, only limits the widgets to three). By going through the transition of *click on widget and drag the widget*, we are now in the following state of Button 1. This state waits for the user to drop the selected widget to get to the next state, Button 2. Here, unlike in the previous state, Button 2 has iteration choice, which the button could edit its property by ‘right click’. The transition of ‘double click’ from Button 2 ends the process, which goes to the Finish State. The same concept applies to the rest of the states available.

Figure 6.2:
 State Transition Network
 for Designing Dialogue



6.4 Screen Design

In this section, we will see how the generic guidelines can be turned into more realistic feature. This is done by designing a proposed screen design of the layout of the GUI builder. Figure 6.3 below is the attempt to impose everything that has been discussed earlier.

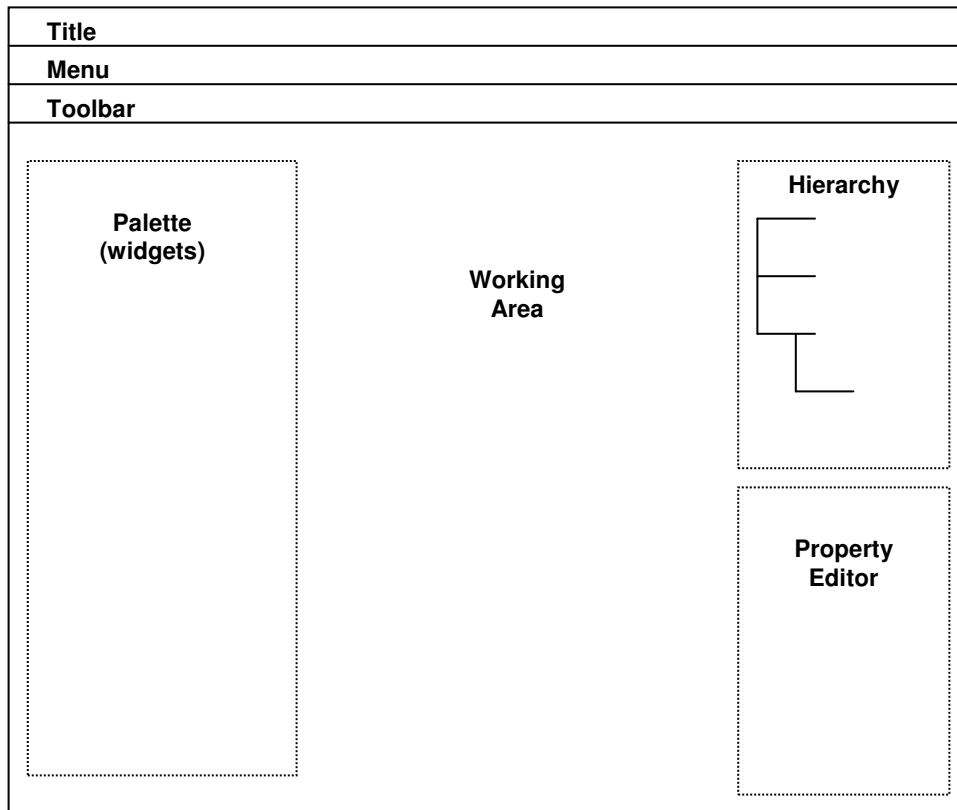


Figure 6.3: The Proposed Screen Design

From the above layout, we could clearly see the distinction between one area to another. By clearly distinguishing their titles differently, will enable the separation of the tasks. This further enables a good organisation of information. The layout of the screen has a very small number of windows. The amount of four is still in the range of the suggested minimal number of windows (3 to 7 windows).

The Menu bar will be consisting of standard options like, File, Edit, View and Help, which these includes all of the basic operations like new, open, save, exit of files, cut, copy, paste, undo functions and help. As we are designing a system for a GUI builder, it

will incorporate other operations like generate dialogue, property editor, and hierarchy. Meanwhile, the toolbar will contain the shortcuts of some operations from the Menu bar, namely the new file, save file, generate dialogue and so on.

The proposed Palette window is the place for all sort of widgets available for the use of designing the dialogues. The widgets will be represented by their own images and names. From here, the widgets can be chosen and be dragged to the wanted position in the working area. The Hierarchy window will contain the tree list of the selected widgets, which the flow of the hierarchy is corresponding to the 'parent-and-child' format.

In the Property Editor window, it will consist of the widgets' attributes. However, the values that will appear in the Property window at one time is belonged to the widget that is currently chosen by the user. The working area will have some kind of grid image as its background to guide the design positioning. Below are the proposed examples for the grid.

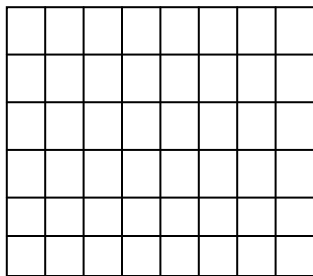


Figure 6.4: Cross Grid

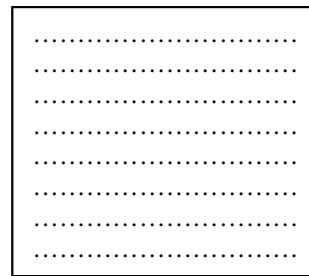


Figure 6.5: Dots Grid

6.5 Conclusion

The direct manipulation concept is very essential in the interactive system, such the GUI builder itself. As the GUI builders are designed to build graphical user interface, it is important to have the WIMP features in the system. The diagrammatic dialogue of the STN has enabled us to recognise the structure of the design dialogue in particular, in a glance. The proposed screen design will be essential in going through the implementation of the GUI builder prototype, which this will be discussed at great length in the next chapter.

Chapter 7: Implementation

The implementation process is derived almost entirely from the previous chapter; the Design. It is important to note that the implementation of a prototype is carried out only as an attempt to show how the usability of the generic guidelines is applied to the GUI builder. The prototype is not in the form of a complete fully functioning GUI builder system that abides the generic guidelines.

The first section of this chapter concerns the technology used in the implementation stage of the GUI builder prototype. The next section will elaborate further on the prototype of the GUI builder interface and on some functionality that one generic GUI builder must have in its system. The latter area will also highlight some views on why the proposed implementation is better than the one that is already existing.

7.1 Technology Used

The prototype was implemented using Microsoft Visual C++ 6.0. Following are the reasons for choosing Microsoft Visual C++ 6.0:

- **Availability**

The Microsoft Visual C++ 6.0 tool is available in every computer from the MSc laboratory. Thus, there is no complication in acquiring the software.
- **Fast Development**

This tool provides Microsoft Foundation Class Library (MFC), which has enabled the development to be carried out in a faster pace. In addition, the provision of online documentation is very efficient, and, the debugger is also useful in the sense that it decreases the development time.
- **Low Risk**

This tool is widely used in today's software development community due to its stability. Thus, this proved the minimal amount of malfunction risks.

7.2 The Prototype

The implementation is kept to minimal and simple as the objective of implementing the prototype is to illustrate the findings from Chapter 5 and 6, which it is mainly concerned the layout interface and certain functionality. In order to focus on how the implementation put the generic guidelines into reality, we separated implementation of the screen interface and the major functionality.

7.2.1 Screen Interface of the GUI builder

The implemented screen interface is developed according to the proposed design from Section 5.3 in the previous chapter. Figure 7.1 below is the implementation of the screen interface.

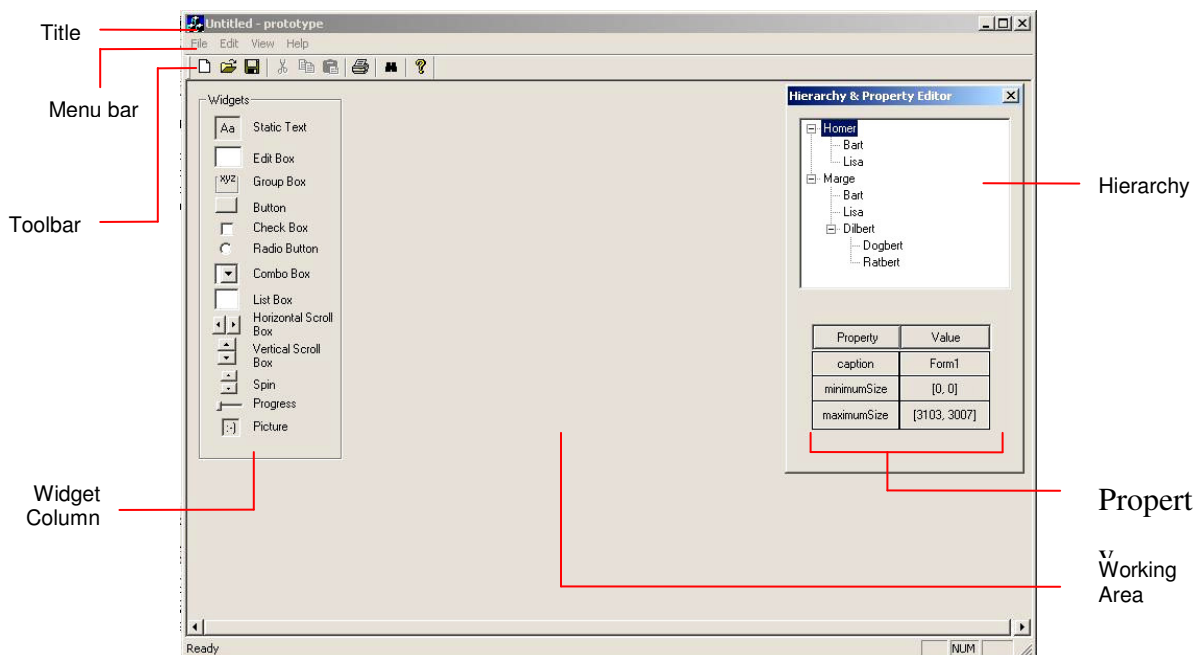


Figure 7.1: Prototype Screen Interface

From the screen interface above, we could see that it applied almost everything from the proposed design. Therefore, it certainly abides the generic guidelines that have been outlined previously. Each area clearly distinguishes their tasks, which this gives a good organisation of information to the users. The title that appears at the top of the window

should refer to the work that is currently done by the user. As there is no work is taking place, the window for the prototype GUI builder is set to 'Untitled' by default.

In order to let the users become easily familiar with the *Menu* bar, it then consists of the usual options like *File*, *Edit*, *View* and *Help*. It differs from any other non-GUI builder tool by including the *Generate*, *Hierarchy* and *Property Editor* as the *View* sub-menus. The list below shows the sub-menus of each option.

- *File*: New, Open, Save, Save As, Print, Print Setup, Recent File, Exit
- *Edit*: Undo, Cut, Copy, Paste
- *View*: Toolbar, Status bar, Generate, Property Editor, Hierarchy
- *Help*: About GUI builder, Help Topics

The toolbar, which offers shortcuts to the users, is constituted of *New*, *Open*, *Save* and *Print* dialogues from the *File* Menu; *Cut*, *Copy* and *Paste* from *Menu* Edit; *Generate Dialogue* from *View*, and *Help* from Help Menu. If we noticed, the *Generate Dialogue* bar's icon (Figure 7.2), is similar to *Find* icon in other windows application. This, however, will not confuse the users, as a short caption appears when the user brings the mouse on top of the icon.

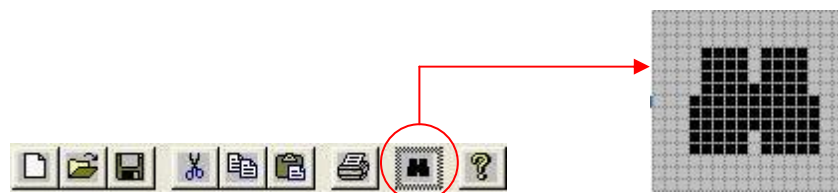


Figure 7.2: Toolbar and Generate Icon

Section 4.1.3 from Chapter 4 has emphasised the consistency of colour usage and positioning. The screen interface uses the same colour as any other windowing system, i.e. greyish colour, to bring familiarity to the users. The positioning is also consistent in a way that the *Title*, *Menu* bar and *Toolbar* apply the 'left-to-right' concept. By having the *Property Editor* and *Hierarchy* windows on the right side of the screen and *Widgets* window on the left, leaves the working area in the middle. This kind of positioning

feasible the design works, as the users could easily see, or aware the 'options' that are available around them. It is believed to give more 'visibility' to the users. This is different from the way the wxwindows, and visaj software had offered. If they do not wish to have the *Property Editor* and *Hierarchy* windows appear on their right, they could simply close it.

The widgets available in the *Widgets* column on the left side of the screen provide images of each component, apart from their names. Thus, this assists the users to select the right widget. This overcomes the problem faced by the students when using the lxb software. In addition, this fulfils another generic guidelines, i.e. the '*image icons that depict the widgets*'. Example from Figure 7.3 illustrates this.

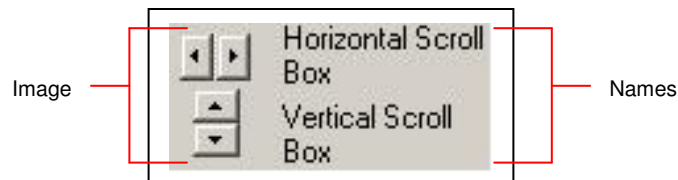


Figure 7.3: Widgets

The *Hierarchy* window from the above screen interface shows an example of a tree-list of an 'American' family. This is merely an attempt to show how the tree-list works. It, however, suppose to correspond to the dialogue design. The same applies to the *Property Editor*. The example consists of *Property* and *Value*, which the users can edit the attributes of the selected widget.

7.2.2 Drag and Drop

One of the generic guidelines stresses on the importance of providing 'easiness' for the users to move the widget to the desired position. The dragging movement must not flicker and a grid should be provided to ensure the accuracy of the positioning. The implementation has been carried out in a way that a square, which represents a widget, can be moved and dragged to the wanted position on the grid. Figure 7.4 below shows the illustration of the 'Drag and Drop' function, which uses cross grid as reference to positioning.

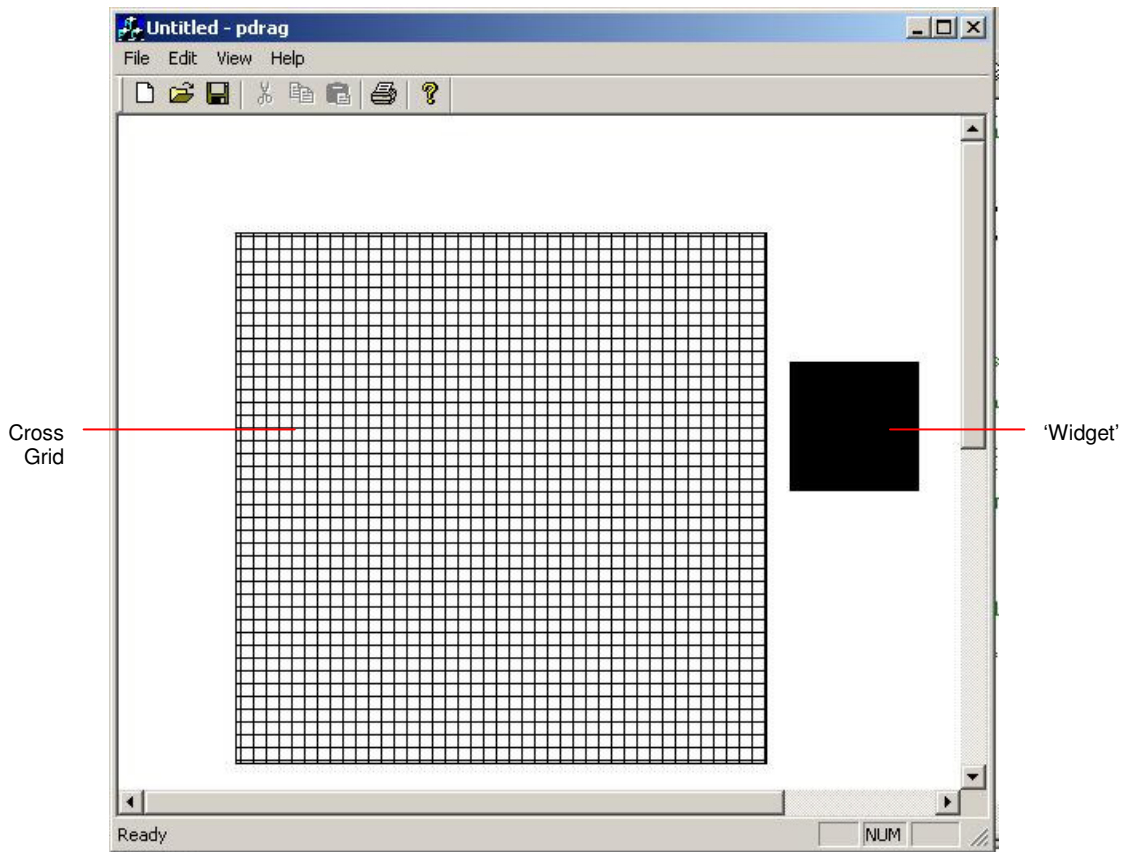


Figure 7.4: Drag and Drop

This certainly is much better than the 'drag and drop' function offered in the JGUIDev, wxwindows and lxb.

7.2.3 Feedback

Conveying a short, yet meaningful message is also one of the important aspects that is emphasised in the generic guidelines. There are many types of feedback. The rapid feedback could be of conveying information message, and warning message. The function, which has been developed, is an attempt to send out a meaningful feedback to the user when the user saves a file with a wrong extension file type. The message informs the user the correct extension type. Figures overleaf illustrate this.

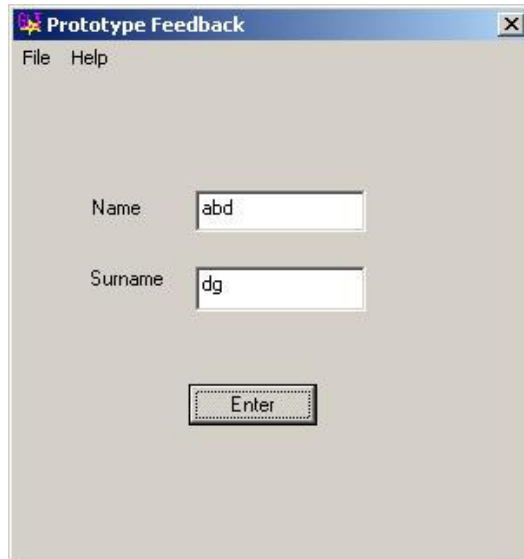


Figure 7.5: Example of Dialogue to be saved

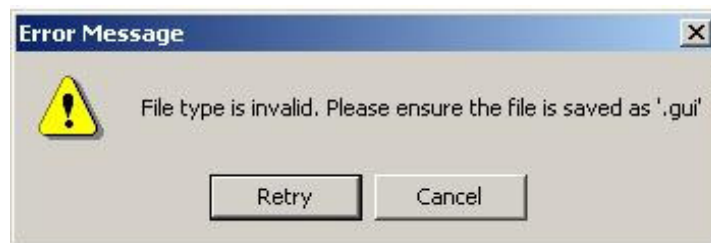


Figure 7.6: Feedback Message

The example above is indeed better than the one we found from JGUIDev software.

7.2.4 Help Documentation

As we noted, Help documentation is very important in guiding the users to use any system. Therefore, the implementation has developed an online documentation, which could be found from Help Menu. The documentation contains Help Topics and Tutorial that enable in assisting the users to start using the GUI builder and whilst working. The development uses the same concept of 'Context-Sensitive Help', which is available in almost all software today. Figure below is the screenshot of the Help Topic.

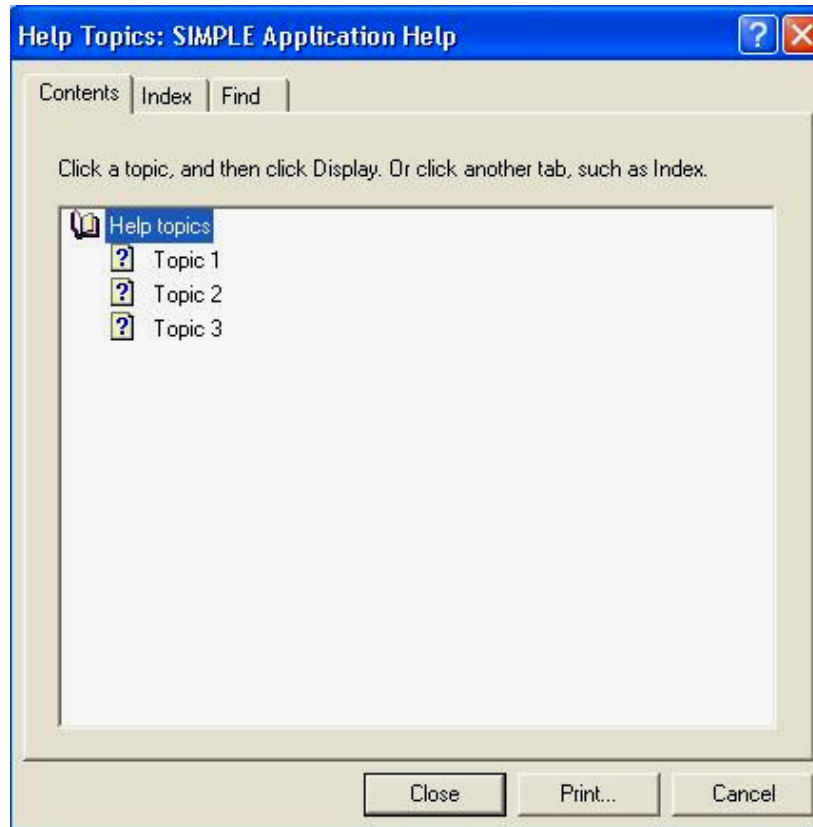


Figure 7.7: Help Documentation

This example is as good as the help manual offered by visaj and Qt Trolltech Designer software, and certainly is better than the rest of the analysed software.

7.3 Conclusion

Microsoft Visual C++ 6.0 has been used for the prototype implementation. In overall, the developed prototype has conformed to almost all guidelines produced in Chapter 4⁸. From this chapter, we have been able to see how the screen interface incorporated the suggested screen design that has been created in the previous chapter, and also, the implementation of some major functionality of the GUI builder. In addition, the implementation has enabled us to see how it overcomes certain problematic areas that have been raised in the analysed software.

⁸ Please refer to Appendix 8 for the Check List table

Chapter 8: Further Work

This chapter consists of three sections. The first section is dedicated to a discussion on how to improve the investigations that leads to the generic guidelines. The following section describes the extension work towards the completion of the existing project. The third section, which is the final section, outlines the further research related to this project.

8.1 Improvements

Due to the limitations of time and cost, the author managed to analyse only five software tools, which each of this software is free and available from the Internet. It is inevitable to avoid from being questioned about the generic guidelines' credibility. Therefore, the most important point that the author thinks could improve the results of the generic guidelines is to considering a larger number of GUI builder software tools. The author believes that the larger the number, the more firm or solid the guidelines would be. This, however, will only leads to a weaker result, if the selection process of choosing the correct type of software tools doesn't take place.

Selection Process

There are two categories, which have been identified earlier in this project, that are important to be considered in the investigation; commercial and research. As there are so many software tools available in today's world, a meticulous selection must be carried out before the investigation could take place. Only those, which are well established and widely used, are worth to be investigated. This is vital especially when selecting the commercial tools. By being selective only on those that are widely used, we are able to discover what criteria the tools have that made them so successfully accepted in the market. From the research point of view, there is one example of software tool that is worth to be taken into investigation (as has been mentioned in chapter 3). It is the Amulet software tool. The Amulet project is run by Brad A. Myers, who is renowned for his research work in User Interface Software, who is currently releasing the latest version of Amulet; the Open Amulet 3.0.

Suggested numbers and Target groups

As we noted earlier, the author has suggested a larger number of software tools should be considered for the investigations. The most realistic number that the author think is affordable to be carried out in the investigations is in the range between 5 to 10 for each category. In the survey operation, the participants should be only of those who have computing background or those who are the developers themselves. This is essential because we need to remember that the GUI builders are normally being used by these target groups, in order for them to develop applications for their software.

8.2 Extension Work

As the main purpose of this project is to come out with generic guidelines for GUI builders, the implementation stage has only focused on developing the prototype, in such a way that it illustrates the generic guidelines in the usability context, rather than in the form of a complete functioning development of a GUI builder system.

Fully functional prototype

Therefore, this project can be extended, in the matter of completion, with the implementation work of a proper and fully functional GUI builder prototype. This can be done either by adhering to the Microsoft Visual C++ development tool, or by using the Java AWT toolkit or its recent version, the Java Swing that has been purposely designed for GUI development. The significant reason of pointing out these technologies is, to remark the existence of styleguides for Windows and Java platforms. Each platform should conforms to its styleguide when developing the GUI builders system in order to bring the best out of each platform, with at the same time abiding to the generic guidelines. The outcome, however, should give similar product as each follows the generic guidelines.

Testing and Evaluation of the builder

Testing and evaluation, then, can be carried out to prove the efficiency of the generic guidelines in the GUI builders. Different approaches should be taken when testing and evaluate the GUI builder. Due to its own *generic* nature, it is very important for us to

consider some issues like what type of designers should design the GUI builder and what type of users should test and evaluate the system.

If we do not carefully identify the type of designers, they can design the GUI builder according to what THEY feel is best, even if they take into account all the generic guidelines. Therefore, the generic guidelines have the chances to be misled and misguided, which this would not prove the efficiency of the guidelines.

The same thing could occur if we wrongly chosen the type of users who can test and evaluate the GUI builder. Different level of expertise would give different results, according to different interpretations. By identifying the target group beforehand, e.g. computing students, we could at the very least, expect the users' reaction towards the generic guidelines.

Therefore, for a start, in order to prove the effectiveness of the generic guidelines, we could design two or three different GUI builders, in which one must conforms to all rules, one does not apply any of the guidelines, and another only takes some of the rules. This, of course, is done after we have selected the type of designers and users. By doing so, we are hoping to see different result performance of the three.

8.3 Future Research

As for the future research regarding to this project, the author would like to suggest refine guidelines, which is called 'generic-yet-specific'. This guidelines differ in a way that it corresponds to the particular type of field or area the GUI builders have the prospect of being used in - generic, yet specific.

The work could be started with the same procedure, which is investigating and analysing the commercial software tools, as well as research tools, that are widely used in the industry. Besides emphasising on their design, it is also important to identify in which area of the tools have been widely used in. Therefore, a pattern of design categorised by their usage can be obtained. As at most of the times, analysing from one's point of view is

not sufficient enough. Thus, a number of surveys from the users of the GUI tools are of importance towards the analysis before the generic guidelines could be established.

Then, the usual software engineering lifecycle can be adapted. Starting off by designing the builders, which this incorporates the findings, and then developing the design by using a suitable programming language. Iteration process of testing, after the developing stage, is inevitable. Then only the users could evaluate the final products. Testing and evaluation processes must adopt the same principles that have been raised in the previous section. Diagram overleaf (Figure 8.1) describes the stages of the process.

Any feedback received from the evaluation must be forwarded to the design stage, according to the area each comes from, as an input, in order to re-design to meet the users' requirements. The iterative is put into a halt, once the builders fulfil their objectives, as well as meet the objectives of usability and interactivity.

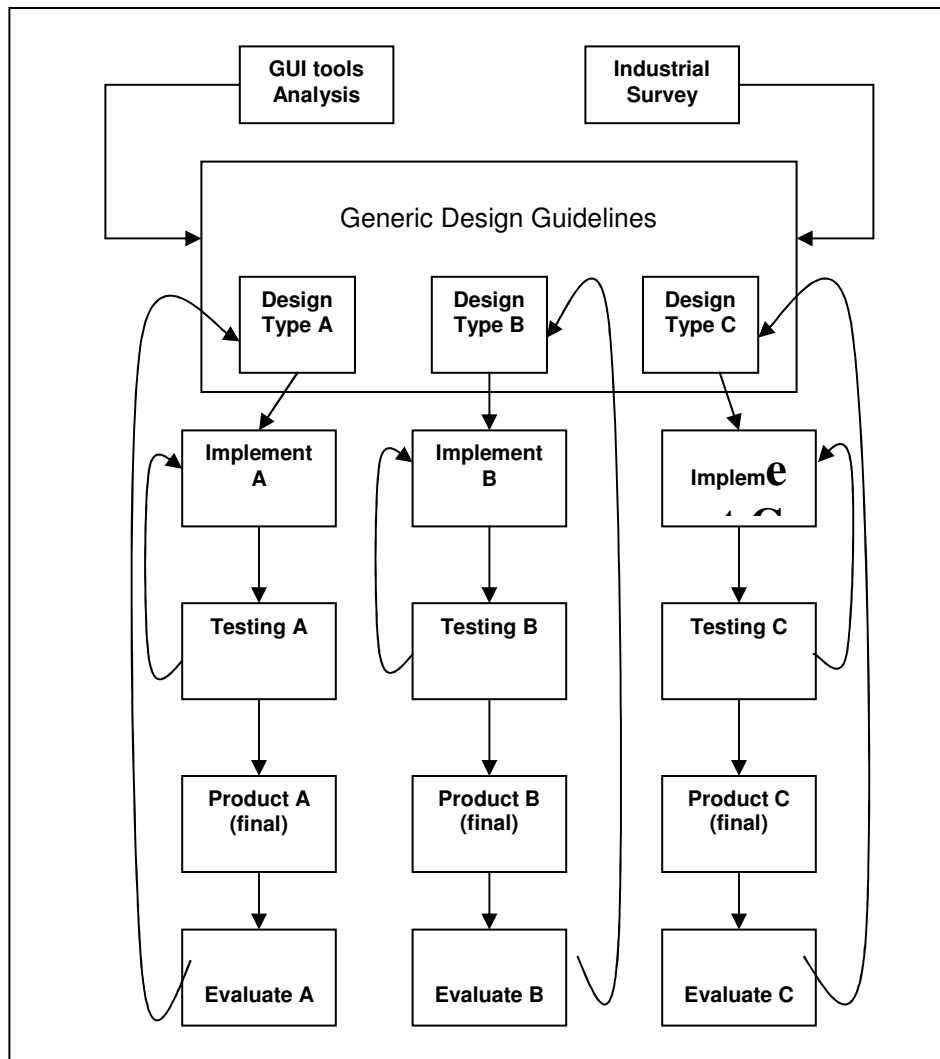


Figure 8.1: Research Plan

Chapter 9: Conclusion

The important goal of this project is to suggest ways of providing 'better' more generic GUI builders. This is achieved by outlining generic guidelines of GUI builder designs in particular, and functionality and overall concept in general. The implementation of a prototype is carried out only as an attempt to show how the usability of the generic guidelines can be applied to the GUI builder, and not in the form of a complete fully functioning GUI builder system that abide the generic guidelines.

The generic guidelines, which are believed to be able to provide better GUI builders, are grouped into three. The first concerns the layout designs of the GUI builder, secondly is about the functionality that the GUI builders must be able to perform, and thirdly is the overall performance each builder must has or must be able to perform.

Investigations have been conducted onto three major areas, in order to derive to the generic guidelines. It started off by performing a literature survey on three significant areas, which are GUIs, GUI builders and the good UI designs. The other area focused on the analysis of various types of GUI builders and generators. The third area concerned with the survey operation carried by 70 students of the second year computing, which the students need to test and evaluate one chosen GUI builder (Ixb software). Two sets of questionnaire have been used to evaluate the software. There are the Questionnaire of User Interface Satisfaction (QUIS) and Computer System Usability Questionnaire (CSUQ).

The results from each area, then, were put together in order to proceed with the derivation process, which this produced the generic guidelines. Amongst the domain subjects found from the results are encompassing the characteristics of a good GUI builder, the GUI builder usefulness and their designs.

To reflect how successful the investigations have been, first we evaluate the reason(s) why the proposed project plan did not go to plan. It is suspected that this is caused by the

survey operation, which has consumed so much time. Furthermore, this operation only took place after the analysis of several software is done, as oppose of working on both of them in parallel (as shown in the proposed project plan). Thus, this led to the lagging of other processes in which they could only be proceeded once the survey is completed. The author feels that is it important to look for the best method of analysing survey results beforehand, in order to guide the author in analysing the results, rather than spending time on working out how and what is the best approach to take. In addition, if the author could approximate the time the analysis normally takes in advance, it would be one of the methods than can avoid this problem from happening.

Secondly, due to the limitations of time and cost, the analysis of various GUI builders and generators have only allowed the investigation to be carried out on the free and available software tools. Therefore, we could say that the generic guidelines produced from the investigation operation could have been better if the things that have been mentioned do not occur. Some suggestions to improve the results have been given in the Further Work chapter.

Further work has reconsidered some significant issues, like type of tools, amount of tools, and users target, which are worth to be contemplated if we want this work to be extended and furthered. Due to the generic nature of the builder itself, the testing and evaluation processes need to consider different approaches.

In summary, the generic guidelines produced, which provide a better GUI builder, is believed to be beneficial to computing field as currently, the research into this particular area is still at its infancy (from the author's experience when undergoing the literature review).

References:

[1] J. P. Chin, V. A. Diehl, and K. L. Norman, *Development of a Tool Measuring User Satisfaction of the Human-Computer Interface*, ACM CHI'88 Proceedings, pp 213-218, 1988 ACM [Online Document]

Available HTTP: <http://lap.umd.edu/lapfolder/papers/cdn.html>

[2] *Computer Usability Satisfaction Questionnaires* [Web site]

URL is accessible from <http://www.hcibib.org/gs.cgi>

[3] *Computer System Usability Questionnaire* [Online Document]

Available HTTP: <http://www.acm.org/~perlman/question.cgi?form=CSUQ>

[4] A. Dix et. al, *Human-Computer Interaction*, Prentice Hall, Second Edition, pp 120, 296 and 444, 1992

[5] H. Evagelos, *Implementation of an Interactive Graphical System to support the learning of basic concepts of programming*, MSc Dissertation, Lancaster University, pp 9-13, 2000

[6] Z. Fei, *GUI Builder Tools* [Online Document]

Available HTTP: http://www.cc.gatech.edu/classes/cs6751_97_winter/Topics/gui-builder/

[7] J. R. Lewis, *IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use*, *International Journal of Human-Computer Interaction*, v.7 n.1 p.57-78, Ablex Publishing, 1995

[8] A. Meyyapan, *Front-end GUI Development using XML Technology*, [Online Document]

Available HTTP: <http://www.planetpublish.com/xmlarena/xap/Thursday/GUI.pdf>

[9] B. A. Myers, *User Interface Software Tools*, Computer Science Department, CMU University, Pittsburgh, PA 15213, (412) 268-5150, August 1994

[10] G. Perlman, *Gary Perlman's Home Page*, [Web site]

www.acm.org/~perlman/index.html

[11] G. Perlman, *Web-Based User Interface Evaluation with Questionnaires* [web site]

Available HTTP: <http://www.acm.org/~perlman/question.html>

[12] *Questionnaire for User Interface Satisfaction*, [Online Document]

Available HTTP: <http://www.acm.org/~perlman/question.cgi?form=QUIS>

[13] QUIS™, *Questionnaire For User Interaction Satisfaction*, [Web site]

Available HTTP: <http://lap.umd.edu/quis/>

[14] B. Shneiderman, *Designing the User Interface Strategies for Effective Human Computer Interaction*, 3rd Edition, Addison-Wesley, March 1998

[15] webopedia, *Graphical User Interface*

Available HTTP:

http://www.webopedia.com/TERM/G/Graphical_User_Interface_GUI.html

[16] whatis?com, *Graphical User Interface*

Available HTTP: http://whatis.techtarget.com/definition/0,289893,sid_gci213989,00.html

[17] X Business Group Inc., *Interface Development Technology*, 3155 Kearney Street, Suite 160, Fremont, CA 94538. (510) 226-1075, 1994

Bibliography:

A. Anjewierden and J. Wielmaker, *Why GUI-Builders are Evil*, [Web site]

Available HTTP: <http://www.swi.psy.uva.nl/projects/xpce/noguibuilder.html>

D. Chapman, *Sams Teach Yourself Visual C++ 6*, Sams Publishing, 1998

P. Gray and R. Took, *Building Interactive Systems: Architecture and Tools*, Springer-Verlag, 1992

J. Hobart, *Principles of Good UI Design*, [Web site]

Available HTTP: http://axp16.iie.org.mx/Monitor/v01n03/ar_ihc2.htm

J. K. Kruglinski, G. Shepherd, and S. Wingo, *Programming Microsoft Visual C++ Fifth Edition*, Microsoft Press, 1998

R. A. Stalker, *The Design and Development of IDIOM-GUI Graphical User Interface for Interactive Design using Intelligent Objects and Models*, Technical Report No 96/211, Ecole Polytechnique, Federale De Lausanne, June 1996

L. C. Tai, *The GUI Toolkit, Framework Page*, [Web site]

Available HTTP: <http://www.free-soft.org/guitool/>

Appendix A: Questionnaire for User Interface Satisfaction Results (a)

Questionnaire for User Interface Satisfaction											
Score	0	1	2	3	4	5	6	7	8	9	NA
Question											
1	VVV	VVVIII	VVVVIII	VVI	VII	I	III				
2	III	II	VVI	III	VV	VIII	VIII	VII	VIII	V	I
3	VVVVVVII	VVIII	VI	VVI		III	II	I			II
4	VV	VVIII	VVIII	VVIII	IIII	V	IIII	I	III		I
5	VIII	VV	VVIII	VVIII	VIII	VIII	II	III		I	I
6	VVI	VVIII	VVVI	VV	VI	V	III	I	I	I	I
7			VI	III	VIII	VV	VIII	VVIII	VVIII	VV	III
8	VIII	I	IIII	VI	VVI	VIII	VVI	V	III	II	VV
9	III	III	VIII	V	VVIII	VVIII	VVIII	VIII	IIII	I	I
10	III	II	VII	VVI	VII	VVIII	VV	VIII	II	III	III
11			III	VII	V	VVIII	VVVI	VVVI	V	VII	II
12		III	V	IIII	VIII	VVII	VVIII	VVII	VII	IIII	II
13	VI	III	V	V	VVII	VV	VIII	VVIII	VII	III	II
14	III	V	VVIII	IIII	VVIII	VVV	V	IIII	II	IIII	V
15	VVII	VIII	VVII	V	VIII	VVII	V	II	I	II	III
16	VVVV	VVVIII	VVII	II	VI	III	I		II		VI
17	IIII	II	IIII	IIII	V	VII	VVI	VVI	VVII	VV	
18	V	VV	VIII	VI	VIII	V	VIII	VI	VVIII	V	
19	II	II	VI	V	VVIII	VI	VVIII	VVII	VV	VVIII	
20	VIII	V	VVIII	VI	IIII	VIII	VV	VVI	IIII		
21	VVVVII	VVIII	VIII	V	VI	II	I				VVIII
22	VVII	VI	VIII	V	VI	VI	I	VI	I	I	VVVIII
23	VVVVVVIII	VVI	VII	VI	III	I	II		I		
24	VVVVVVVVIII	VVIII	VII	VI	I		II				I
25	III			I	V	VI	II	V	VII	VVIII	VVVVIII
26	VVVVIII	VV	VVII	VVIII	I	I	V	V	III	I	
27	VVII	VVIII	VVI	VV	VIII	VVIII	I	VII	I	II	

Appendix B: Questionnaire for User Interface Satisfaction Results (b)

Questionnaire for User Interface Satisfaction													
Score		0	1	2	3	4	5	6	7	8	9		NA
OVERALL REACTION TO THE SOFTWARE		(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)		(%)
1	terrible	21.4	20.0	27.2	15.7	10.0	1.4	4.3	0.0	0.0	0.0	wonderful	0.0
2	difficult	5.7	2.8	15.7	4.3	14.3	12.9	12.9	10.0	12.9	7.1	easy	1.4
3	frustrating	45.7	18.6	8.6	15.7	0.0	4.3	2.8	1.4	0.0	0.0	satisfying	2.8
4	inadequate power	14.3	20.0	20.0	20.0	5.7	7.1	5.7	1.4	4.3	0.0	adequate power	1.4
5	dull	12.9	14.3	20.0	20.0	11.4	11.4	2.8	4.3	0.0	1.4	stimulating	1.4
6	rigid	15.7	20.0	22.9	14.3	10.0	7.1	4.3	1.4	1.4	1.4	flexible	1.4
SCREEN													
7	inconsistent	0.0	0.0	8.6	4.3	11.4	14.3	11.4	18.6	12.9	14.3	consistent	4.2
Reading characters on the screen													
8	not at all	12.9	1.4	5.7	8.6	15.7	11.4	15.7	7.1	4.3	2.8	very much	14.2
Highlighting simplifies task													
9	confusing	5.7	4.3	11.4	7.1	12.9	18.6	20.0	11.4	5.7	1.4	very clear	1.4
Organization of information													
10	confusing	4.3	2.8	10.0	15.7	10.0	20.0	14.3	11.4	2.8	4.3	very clear	4.2
Sequence of screens													
TERMINOLOGY AND SYSTEM INFORMATION													
11	inconsistent	0.0	0.0	4.3	10.0	7.1	12.9	22.9	22.9	7.1	10.0	consistent	2.8
Use of terms throughout system													
12	never	0.0	4.3	7.1	5.7	11.4	17.1	18.6	17.1	10.0	5.7	always	2.8
Terminology related to task													
13	inconsistent	8.6	4.3	7.1	7.1	17.1	14.3	11.4	12.9	10.0	4.3	consistent	2.8
Position of messages on screen													
14	confusing	5.7	7.1	18.6	5.7	12.9	21.4	7.1	5.7	2.8	5.7	clear	7.1
Prompts for input													

15 Computer informs about its progress	never	17.1	11.4	17.1	7.1	11.4	17.1	7.1	2.8	1.4	2.8	always	4.3
16 Error messages	unhelpful	28.6	25.7	17.1	2.8	2.8	8.6	4.3	1.4	0.0	2.8	helpful	8.5
LEARNING													
17 Learning to operate the system	difficult	5.7	2.8	5.7	5.7	7.1	10.0	15.7	15.7	17.1	14.3	easy	0.0
18 Exploring new features by trial and error	difficult	7.1	14.3	11.4	8.6	11.4	7.1	11.4	18.6	12.9	7.1	easy	0.0
19 Remembering names and use of commands	difficult	2.8	2.8	8.6	7.1	12.9	8.6	12.9	17.1	14.3	12.9	easy	0.0
20 Performing tasks is straightforward	never	12.9	7.1	18.6	8.6	5.7	11.4	14.3	15.7	5.7	0.0	always	0.0
21 Help messages on the screen	unhelpful	31.4	18.6	11.4	7.1	8.6	2.8	1.4	0.0	0.0	0.0	helpful	18.6
22 Supplemental reference materials	confusing	17.1	8.6	11.4	7.1	8.6	8.6	1.4	8.6	1.4	1.4	clear	25.7
SYSTEM CAPABILITIES													
23 System speed	too slow	55.7	15.7	10.0	8.6	4.3	1.4	2.8	0.0	1.4	0.0	fast enough	0.0
24 System reliability	unreliable	62.9	12.9	10.0	8.6	1.4	0.0	2.8	0.0	0.0	0.0	reliable	1.4
25 System tends to be	noisy	4.3	0.0	0.0	1.4	7.1	8.6	2.8	7.1	10.0	18.6	quiet	40.0
26 Correcting your mistakes	difficult	32.9	14.3	17.1	12.9	1.4	1.4	7.1	7.1	4.3	1.4	easy	0.0
27 Designed for all levels of users	never	17.1	12.9	15.7	14.3	11.4	12.9	1.4	10.0	1.4	2.8	always	0.0

Appendix C: QUIIS Results Analysis

Questions	Results
<p>Section 1: Overall reaction to the software</p>	<ul style="list-style-type: none"> • From the survey, the students felt that the software is terrible, as the system is not very user friendly. This could be seen from the percentage received, 68.6%, which this constituted the percentages of scores 0, 1 and 2. • The software is neither difficult to use, nor it is easy. The distribution of the percentage ranged from 0 to 9 is about the same. From the comment, it was difficult to use due to its lack of documentation, and even once learnt, it is simply frustrating to use. The least percentage received was 1.4% by the NA option, which this means a few students thought this item does not applicable to the software. • From the students' point of view, the software is very frustrating, rather than satisfying. Most of the higher percentage were obtained by score 0, with 45.7%, and is followed by score 3 with 15.7%. The reason being is, the system sometimes crashes and hence, lost the work. Nevertheless, 2.8% of the students did think that this item is not applicable to the particular software. • The software is also thought to be inadequate in power. The survey supports this fact by showing the obtained scores 1, 2 and 3 as the highest percentage of 20% each. Despite this fact, the Ixb is believed to be able to produce any user-interface that can imagine, although it can be proved difficult. • Between dull and stimulating, the weight of the percentage is leaned towards the 'dull' criteria. The system should have used graphics to attempt to display widget interactions. Only as little as 1.9% of the students think the software is stimulating by scoring 9, as some students find it quite exciting when trying to create a beautiful interface. • The software is also thought to be rigid, rather than to be flexible. By looking at the results, 22.9% has been allocated at score 2 and is followed by score 1 with 20.8%. From the comments received, it is thought to be not flexible, as the system has no ability to link between dialogue boxes.
<p>Section 2: Screen 2.1 Reading characters on the screen</p>	<ul style="list-style-type: none"> • The consistency of reading characters on the screen is said to be quite consistent. The reason being is, from score 4 to 9, each has obtained above 10%, and there scores 0 and 1 remained zero. Nevertheless, there is 4.2% of students' thought these items is not applicable.
<p>2.2 Highlighting simplifies task</p>	<ul style="list-style-type: none"> • The second criteria, highlighting simplifies task, shows a moderate opinion, as each the scores from 0 to 9, has about the same percentage. It is worth to note here, that quite a significant number of students thought this item is not applicable (NA), with the percentage of 14.2%. From some of the comments received, it has been noticed that when highlighted, object text disappeared and often makes an item illegible. On the other hand, highlighting is a very important feature as to know which frame or field in the

	Resource Editor the users enter the values.
2.3 Organization of information & 2.4 Sequence of screens	<ul style="list-style-type: none"> • The following items, organisation of information and sequence of screens have also shows a moderate percentage. Both items are neither confusing nor they are very clear. It is worth to note that for the latter criteria, there was 4.2% of the students, opted for the NA. This means, they thought the sequence of screens is not applicable to be answered. Points that have been raised are the screens need to be reorganised every time application starts and there was no logical sequencing between screens.
Section 3: Terminology and System Information 3.1 Use of terms throughout system	<ul style="list-style-type: none"> • Scores 6 and 7 received the highest percentage, with 22.9% each, and 12.9%, the second highest goes to score 5. From these scores' performances, it can be concluded that the use of terms is consistent throughout system. Despite the fact the same terms always used throughout the system, the terms were found ambiguous in the widget attributes list and within the error messages.
3.2 Terminology related to task	<ul style="list-style-type: none"> • The distribution of the percentages amongst the scores is about the same, which the percentage varies from 0.0% to 18.6%. It shows that the system provides moderate terminology related to task.
3.3 Position of messages on screen	<ul style="list-style-type: none"> • This criterion also shows a moderate opinion of the students about the consistency of this item. Owing to the fact that there is no prominent percentage that stood out among them to give a firm criterion. Some students, however, did pointed out the inconsistency by mentioning the messages sometimes in the console and sometimes as pop-us.
3.4 Prompts for input	<ul style="list-style-type: none"> • The highest percentage is received by score 5 (21.4%), and is followed by score 2 (18.6%). Therefore, prompts for input cannot be imposed as confusing or clear. The reason being is, there is no dominant percentage(s) that could lead the criteria towards left or right anchor. It is said that the input was not clear, as the default values are often so poor that the users always need to change them each time a new widget is added. On the other hand, in the Resource Editor, it was clear in the essential feature required to develop a user interface. By looking at the variation of the percentage from score 0 to 9: 2.8% to 21.4%, the NA option has received quite a significant votes from the students with 7.1%. This group of students believed that this item is not applicable to lxb
3.5 Computer informs about its progress	<ul style="list-style-type: none"> • Between never and always, quite a number of students agreed on never, with 17.1% on score 2. Whilst on the other end, score 9 only received 2.8%, 1.4% on score 8 and 2.8% on score 7. Instead of informing about its progress, the program tells what is happening in console.
6 Error messages	<ul style="list-style-type: none"> • 28.6% on score 0 is the highest percentage, which is then followed by score 1 with 25.7%. It is right to say that the error messages were unhelpful as opposed to helpful. The students commented that the messages were incomprehensible and only appeared after the system had crashed, thus making it impossible to try and correct mistakes, plus, the messages suggest no way to solve it. 8.5% of the students made a significant performance to the

	variation, by voting this item as NA.
Section 4: Learning 4.1 Learning to operate the system	<ul style="list-style-type: none"> From the pattern of the percentages of the scores, each of the scores from 5 to 9 received above 10.0%. Thus, the majority of the students felt it was easy in learning to operate the system.
2 Exploring new features by trial and error	<ul style="list-style-type: none"> By looking at the percentage of the scores from 0 to 9, the percentages' range from 7.1% to 18.6%. If we divide the range into two: 0 to 4 and 5 to 9, the amount of two are about the same. Therefore, exploring new features by trial and error, is neither difficult nor easy, it is moderate. Some of them thought that the simplicity of the system helps a lot to that direction and some other agreed except when it crashes.
4.3 Remembering names and use of commands	<ul style="list-style-type: none"> It is quite easy to remember the names and use of commands, by referring to the survey. This is due to the fact that the range of the percentage of the scores that runs from score 5 to 9 is much higher than the other end.
4.4 Performing tasks is straightforward	<ul style="list-style-type: none"> Performing tasks is almost never straightforward if we looked at the percentage. Score 1 received 12.9%, whilst score 9 receives none. The highest percentage goes to score 2 with 18.6%.
4.5 Help messages on the screen	<ul style="list-style-type: none"> Majority of the students' felt that the help messages on the screen was unhelpful. From the survey, the highest percentage was received by score 0 with 31.4%, and by the look of it, the percentage increased from higher to lower scores that shows the left anchor is more dominant. Score 7, 8 and 9 did not received any votes from the students, and they remain 0.0%. Of the number of students who did this survey, 18.6% considered this item as not applicable (NA).
4.6 Supplemental reference materials	<ul style="list-style-type: none"> The left anchor, with the confusing adjective, has received quite a high percentage if compared to the other side of the anchor. The highest 17.1% goes to score 0, whilst the least, 1.4% goes to score 8 and 9. Therefore, the supplemental reference materials was quite confusing perceived by the students. An interesting fact about this item is, 25.7% (the highest percentage amongst all), chose this item to be not applicable (NA).
Section 5: System Capabilities 5.1 System speed	<ul style="list-style-type: none"> The students thought that the system speed was too slow. The 55.7% is the highest percentage that goes to score 0. This supports the claim that has just been made. From the students' observations, the system had a terrible performance as they had to regularly wait a few minutes after clicking on certain change, and when the diagram becomes bigger, the amount required for alterations or addition on the diagram is also becomes larger.
5.2 System reliability	<ul style="list-style-type: none"> Over 60% of the students' felt that the system is unreliable by looking at score 0, which has received the highest percentage amongst all with 62.9%. Scores 7, 8 and 9 received 0.0% that even proves that system in not reliable. The thing that most concerned the students was, once the system crashes, there was no recovery of the files.
5.3 System tends to be	<ul style="list-style-type: none"> By looking at the pattern of the percentage received, 18.6% of the students thought the system tends to be quiet (score 9). Only 4.3% think the systems tends to be noisy (score 0). Overall, the

	<p>system can be said to be quiet as most of the percentages were distributed greater at the right anchor of the scale. However, as much as 40.0% of the students chose this item to be not applicable (NA).</p>
5.4 Correcting your mistakes	<ul style="list-style-type: none"> • The highest percentage is 32.9% at score 0, which is then followed by 17.1% at score 2. Therefore, over half of the students felt that it was difficult to correct their mistakes. The reasons being are, there were no editing facilities, and no keyboard shortcuts and in some cases does not allow certain operations on the diagram. Only 1.4% thought it was easy.
5.5 Designed for all levels of users	<ul style="list-style-type: none"> • The system is never designed for all levels of users. That is what the percentage claimed to convey by looking at the scores' percentage. Towards the left anchor, the percentages are much higher compared to the right anchor, with 17.1% at score 0, 12.9% at score 1, 15.7% at score 2. Whilst on the other hand, score 9 only got 2.8%, score 8 with only 1.4%.

Appendix D: Computer System Usability Questionnaire Results (a)

Computer System Usability Questionnaire									
Score	0	1	2	3	4	5	6	7	NA
Question									
1		VVV	VIII	VV	V	VV	VIII	VI	
2		VIII	VVII	VII	V	VV	VVIII	VII	
3		VVVVI	VVVII	VI	VIII	VII	III		I
4		VVVVVVV	VVIII	VIII	III	II		I	
5		VVVVVV	VVV	VIII	VI	II	II		
6		VVVV	VVIII	VVI	VVI	III	IIII		
7		III	VII	IIII	VIII	VVIII	VVVVI	VII	
8		VVVI	VVVI	VI	VVI	VI	VI	II	
9	III	VVVVVVII	VVIII	VVIII	IIII	II			
10	I	VVVVVV	VVVII	IIII	III	II	I		
11	IIII	VVVVII	VVVI	VV	VIII		II		I
12	IIII	VVVIII	VVV	VVIII	VII	III		I	
13	VII	VIII	VVI	V	VV	VVIII	V	III	
14	VI	VIII	VVIII	VV	VV	VIII	V	II	
15	II	VII	VI	VVII	VV	VVVI	VV		
16	I	VII	VV	VV	VVIII	VVII	VIII	I	
17		VVVIII	VI	VVV	VVII	V	VI		I
18		VVIII	VVVVIII	VVIII	V	III	IIII	I	
19		VVVVIII	VVVVV	V	IIII	IIII	I		

Appendix E: Computer System Usability Questionnaire Results (b)

Computer System Usability Questionnaire												
	Rate	(Strongly disagree)	0	1	2	3	4	5	6	7	(Strongly agree)	NA
Question			(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)		(%)
1	Overall, I am satisfied with how easy it is to use this system		0.0	23.8	14.3	15.9	7.9	15.9	12.7	9.5		0.0
2	It was simple to use this system		0.0	12.7	19.1	11.1	7.9	15.9	22.2	11.1		0.0
3	I can effectively complete my work using this system		0.0	33.3	27.0	9.5	12.7	11.1	4.8	0.0		1.6
4	I am able to complete my work quickly using this system		0.0	55.5	22.2	12.7	4.8	3.2	0.0	1.6		0.0
5	I am able to efficiently complete my work using this system		0.0	47.6	23.8	12.7	9.5	3.2	3.2	0.0		0.0
6	I feel comfortable using this system		0.0	31.7	22.2	17.5	17.5	4.8	6.3	0.0		0.0
7	It was easy to learn to use this system		0.0	4.8	11.1	6.4	12.7	20.6	33.3	11.1		0.0
8	I believe I became productive quickly using this system		0.0	25.4	25.4	9.5	17.5	9.5	9.5	3.2		0.0
9	The system gives error messages that clearly tell me how to fix problems		4.8	50.8	20.6	14.3	6.3	3.2	0.0	0.0		0.0
10	Whenever I make a mistake using the system, I recover easily and quickly		1.6	55.5	27.0	6.3	4.8	3.2	1.6	0.0		0.0
11	The information (such as online help, on-screen messages, and other documentation) provided with this system is clear		6.3	34.9	25.4	15.9	12.7	0.0	3.2	0.0		1.6
12	It is easy to find the information I needed		6.3	30.2	23.8	22.2	11.1	4.8	0.0	1.6		0.0
13	The information provided for the system is easy to understand		11.1	14.3	17.5	7.9	15.9	20.6	7.9	4.8		0.0

14	The information is effective in helping me complete the tasks and scenarios	9.5	14.3	20.6	15.9	15.9	12.7	7.9	3.2	0.0
15	The organization of information on the system screens is clear	3.2	11.1	9.5	19.0	15.9	25.4	15.9	0.0	0.0
16	The interface of this system is pleasant	1.6	11.1	15.9	15.9	22.2	19.0	12.7	1.6	0.0
17	I like using the interface of this system	0.0	28.7	9.5	23.8	19.0	7.9	9.5	0.0	1.6
18	This system has all the functions and capabilities I expect it to have	0.0	22.2	36.6	20.6	7.9	4.8	6.3	1.6	0.0
19	Overall, I am satisfied with this system	0.0	38.2	39.7	7.9	6.3	6.3	1.6	0.0	0.0

Appendix F: Analysis of CSUQ results

Question	Results
1. Overall, I am satisfied with how easy it is to use this system	Over half of the students (54%) disagree with this statement. 23.8% of the 54% contributed to score 1. One of the negative aspects received is due to the fact that the system is not having undo function which makes it harder to use. Only 9.5% were strongly agree with the easiness of used.
2. It was simple to use this system	From the range 0 to 7, 57% of the students scored 4, 5, 6 and 7, which this shows the majority of them agreed that lxb was simple to use. Of the 57%, however, only 11.1% scored 7 (strongly agree), which this is suspected due to the fact that some constraints were raised when using this system. This, therefore, held back 22.2% of the students' opinion, which preferably to give score 6 instead. None of the students strongly disagree (score 0) with this statement.
3. I can effectively complete my work using this system	The amount of 69.8% of the students disagreed with this item (by scoring 1, 2 and 3). One of the negative aspect that have been pointed out is, without the bugs, which this could be due to reason that the program is not yet finished, they would of been able to complete their work efficiently. Other aspect is, the system disallowed the completion of certain tasks, such as the co-existence of two push buttons in the same dialog box. For score 4, 5 and 6, the percentages are 12.7%, 11.1% and 4.8% respectively. Of these percentages, the students found that they were able to complete the work, but not effectively and the system actually does everything they require, just not that well. Score 7 scored 0 percentage, and 1.6% thought this item is not applicable to be answered.
4. I am able to complete my work quickly using this system	From score 0 to 7, 55.5% scored 1, 22.2% scored 2, and 12.7% scored 3. Thus, this strongly shows that the system is slow. Having analysed the negative aspects the students have raised, the system is reckoned to be slow, even when doing the most mundane of tasks. This might be due to its bad performance or reliability, as the system often crashes.
5. I am able to efficiently complete my work using this system	Over 80% of the students felt this statement is less true. 47.6% scored this statement 1, whilst only 3.2% of them scored 5 and 6 each. Even though the completion is preferable, the efficiency just does not come into it. In addition, it is not that efficient as it is very time consuming. Furthermore, the unreliability, rigidity and slow reaction-speed of system reduced efficiency of the task completion.
6. I feel comfortable using this system	Just over 70% of the percentage scored 1, 2 and 3, which clearly shows that they do not feel comfortable using the system. There was no percentage towards score 7, which means there is none of them think that this item is fully agreeable. Some of them felt the system is temperamental and extremely frustrating. Also, due to the fact that this system is often crashing, the system often losses the users' work.

7. It was easy to learn to use this system	Majority of the students agreed with this item. The largest percentage, 33.3% scored 6, whilst the least 4.8% (emitting the 0%), scored 1. Nevertheless, some felt that those users who are not of computer scientists or computing student would not find it easy to learn. Some observed that it has a steep learning curve, but with perseverance, it is easy to use.
8. I believe I became productive quickly using this system	The highest percentage goes to score 1 and 2 with 25.4% each. This shows that over half of the students do not agree with this statement. The least percentage is 3.2% that scored 7. Therefore, the users became productive only once the basics were learnt, and not until the system began to crash and lost the work.
9. The system gives an error messages that clearly tell me how to fix problems	From the score 0 to 7, over 90% of the percentage goes towards scores 0, 1, 2 and 3 with the highest percentage; 50.8% scored 1. There was no contribution to score 6 and 7. The lowest percentage was 3.2% to score 5. It has been noted that the error messages received do not offer any resolution to problems nor any explanation of how to solve errors is given. From the comment, it is found that there was only one error message that proved helpful. Other negative aspect that has been raised is how easy to miss the errors as they appear in the terminal window. They should have appeared as dialog boxes.
10. Whenever I make a mistake using the system, I recover easily and quickly	Score 1 received 55.5%, which this shows that most of them less agree with this item. Thus, the majority does not recovery easily and quickly whenever they make a mistake. 1.6% is the lowest percentage, which contributed to score 0 and 6. This system lacks from editing facilities like undo, and normally it results in a crash.
11. The information (such as online help, on-screen messages, and other documentation) provided with this system is clear	Score 1 received 34.9%, which this is the highest percentage of the rest, followed by 25.4% that contributed to score 2. Therefore, many disagree with this statement. The least percentage received is by NA (not applicable), 1.6%, which this means that these students thought this item is not applicable. Amongst the comments given by the students are, the help menu was not available, i.e. empty, and the online help was not helpful. Many found the tutorial was good, however, this tutorial was not embedded in the system.
12. It is easy to find the information I needed	Score 1 and 2 totalled up 54.0% with 30.2% (the highest) and 23.8% respectively. This shows that most of them find it difficult to find the information. Only 1.6% of them thinks it is easy, by scoring 7 (the least). The students only could rely on the information provided by the lecturer, but not from elsewhere, e.g. web sites.
13. The information provided for the system is easy to understand	The highest percentage is 20.6% which contributes to score 5, whilst the least goes to score 7 with 4.8%. The interesting fact that is worth mentioning is, from score 0 to 3, the total percentage is 50.8% and from score 4 to 7 is 48.2%. This concludes that half of them agreed and half do not. They do not find it easy as most words are in jargon or not present at all. But this compensates with the information provided by the lecturer.
14. The information is	From the score, the highest percentage is 20.8% which is

effective in helping me complete the tasks and scenarios	received by score 2, whilst the least, 3.2% goes to score 7. From overall weight of percentages, much of them were contributed towards the disagree stance. This is due to the fact that there was no information provided in the system. The one that really helped was given by the lecturer, which fairly well explained all the basic functions.
15. The organisation of information on the system screens is clear	25.4% of the students agree with this statement by scoring 5. 0 score received the least percentage by the students with 3.2%. Of the 3.2%, one of the comments received is the ambiguous window, which are often offset, requiring the dragging back into the centre of the screen.
16. The interface of this system is pleasant	From score 0 to 7, it could be summarised that the percentage is slightly greater towards the agreeable stance (4 to 7). However, some notes are worth considering improving the interface. Some of the comments are, there were too many windows to organise, there was a lack of icons and some graphical presentation was confusing.
17. I like using the interface of this system	1.6% think this item is not applicable. Most of the students do not agree with this item just by looking at the weight of the percentage. 28.7% scored 1, and this is followed by score 3, 23.8%. The disagreement is due to the fact that system crashes easily. A suggestion is made onto providing some icons to give some ideas what buttons did.
18. This system has all the functions and capabilities I expect it to have	Score 2 has the highest percentage amongst all, with 36.6%. This is followed by score 1 with 22.2% and score 3 with 20.6%. These prove that most of the students do not agree with this particular item. Some of the comments say that the system cannot link dialog boxes, and, short of an undo function and help information. The least percentage seemed to agree with this statement with 1.6%, which sees the system, seemed to be able to do nearly everything the users required.
19. Overall, I am satisfied with this system	Score 2 with the highest percentage of 39.7%, which is then followed by score 1 with 38.2%, prove that the students are not really satisfied with this system. Although the system does the job, it is slow and very time consuming. Only 1.6% of the students was quite satisfied with the system by scoring 6.

Appendix G: Negative and Positive Aspects of QUIS and CSUQ

1. Negative Aspects

Type	Criteria
1. Performance	<ul style="list-style-type: none"> • flaky performance • inconsistent behaviour • unable to link dialog boxes • have to insert a frame for each item in the dialog boxes • system sometimes corrupts user's data files • inconsistency and confusion with regards to saving work (it does not save the work) • unstable, i.e. crashes regularly for no apparent reason and will not reopen file • time consuming, which the system is too (extremely) slow when making changes and whilst with multiple boxes on screen • difficult to understand and explore • unreliable storage of files • the unpredictable nature of the system when moving widgets around • inflexibility • the work was destroyed when an attempt to delete a frame was taken • system is very difficult in correcting mistakes • no prevention of invalid inputs, as the user only find out when there is an attempt to load the saved file that a previous action was invalid, and the file fails to load
2. Features	<ul style="list-style-type: none"> • no cut and paste feature • unable to undo or revert errors which loses work • no editing abilities • application window had to be completely redrawn everytime the user changed any widget name, or added/removed a widget • can only create one screen within each file, so have to have several files • only allow designer to create very few frames in one dialog boxes • incomplete, so some functionality is missing (too few functions) • cannot change associations easily in the application view • only works on UNIX • cannot right click • the application does not appear to be able to support the functionality it offers, such as implementing drop down menus causes program to crash when switched to 'play mode' or makes files unopenable if saved and then closed / reopened • too few functions (lack of range of components) and capabilities • some things are not intuitive • placing widgets effectively is difficult
3. Layout / Design	<ul style="list-style-type: none"> • interface is a little tricky to learn at first (need some time to learn) • poorly designed, i.e. the interface is dull, basic, not very clear and difficult to use as the three windows take up desktop space

	<ul style="list-style-type: none"> • windows never keep to their resized position • too many windows, and they are confusing • initial layout of screens is confusing, i.e. the user always has to readjust them as the start up dialog boxes are small • data is spread across too many dialog boxes • the interface has too much text and they are very difficult to read • difficult to find the required widget attributes (confusing widgets attributes - nonsensical variable names and not in alphabetical order) • changing fonts resets the size of buttons • finding way around can be awkward • the layout of the Resource Editor is both hard to understand and confusing • windows are difficult to use • poor clarity of options in the three boxes
4. Online Help / Documentation	<ul style="list-style-type: none"> • no help feature for querying different features and options • lack of supporting material and information • no clear explanation of how to use the system • poor levels of online help with regards to its different features
5. Feedback	<ul style="list-style-type: none"> • does not prompt the user when there is an error • no prevention of invalid inputs • no warning when quitting if document not saved first • each change to a widget property needs to be confirmed before moving on to the next, otherwise the change is lost • no warning about overwriting previous files when saving • lack of useful error messages, i.e. did not have much of an explanation • unhelpful error messages with no chance to fix errors • cant be sure work has been saved (no related information is provided)

2. Positive Aspects

1. Learning	<ul style="list-style-type: none"> • easy to use to create simple, stand alone dialogue boxes • relatively easy to learn (without advanced functions) • reasonably straightforward to use • use simple action verbs throughout • simple descriptions • simple to make changes to existing boxes • fairly intuitive • the learning time required is short • simplicity in creating boxes and labels
2. Features / Functionality	<ul style="list-style-type: none"> • it can produce some good simple dialog boxes • easy to correct mistakes • the ability to generate C code with ease for portability • frequent updates and refresh on work • easy reference to commands • easy to see changes as they are made

	<ul style="list-style-type: none"> • auto saves a temporary version as the users are working on a project • components in a dialog box can be moved around with relative ease • able to change the colour • can organise the information to suit the user • provision of recorded plan (family tree) of what you have done and changed in the program, therefore, helpful in remembering structure • lots of options and widgets available • able to click and drag and drop • can see the dialog boxes being built as you go along • consistent presentation of screens throughout • the program appears to have a lot of features allowing for advanced users to meet their initial designs quite easily while maintaining ease-of-use for new users • dialog boxes components could be moved around with relative ease • good functionality • large number of properties to create very flexible interfaces • common tasks easily performed • Resource Editor is a good way of editing an item properties • able to change the title • able to resize windows multiple ways
3. Layout / Interface	<ul style="list-style-type: none"> • reasonably good layout of information • organisation of information is quite good • use of mouse is easier comparing to the ability to change with a keyboard • all information is constrained so the users cannot go outside the boundaries of the language the users are in. thus, allowing the users not to produce something which is not possible • can position windows however the users like • creates realistic dialog boxes • consistent layout of screens • logical - the program has a very logical flowchart/display based layout • split screens for logical designation of tasks • graphical representation of GUI construction - shows how elements fit together • family tree diagram helpful for remembering structure • widget palette interface is very clear • clear layout and simple interface and has a fast interface to use • windows are resizable • ability to see different aspects of design is useful • consistent interface • window layout is clear when arranged so that all information can be seen • visible
4. Concept	<ul style="list-style-type: none"> • the concepts within the program are used consistently, which the elements within frames and other concepts are not difficult to grasp and utilise • some reasonable direct manipulation principles applied e.g. selecting widgets from palette

	<ul style="list-style-type: none"> • separation of like tasks (different tasks are easy to carry out) • reach or ability to manipulate objects • logical organisation • better than programming each interface item manually • free
5. Feedback	<ul style="list-style-type: none"> • easy to see result of changes • when it crashes, a detailed error message is displayed in the shell
6. Performance	<ul style="list-style-type: none"> • when system is running fine, the output is quite fast (speed)

Appendix H: Prototype Implementation Check List

Criteria	Action (Done)
Minimal number of windows (3 to 7)	√ - 3 windows
Give each window a title, to distinguish tasks	√
Moderate amount of text	√
Organisation of information	√
Hierarchy / Flowchart (tree list)	√ (included in the prototype)
Image icons that depict the widgets	√
Grid	√ (included in the prototype)
User-friendly	Not applicable
Easy to learn and use	√ (familiarity concept and provision of Help)
Stable	√ (uses Visual C++)
Stimulating	-
Flexible	-
Fast	√
Consistent	√ (positioning and colour)
Able to link dialogs	-
Save work with confirmation	√
Easy to move widgets around	√
Easy to correct mistakes	√ (feedback message and undo function)
Cut and paste	√ (included in the prototype)
Undo or revert	√ (included in the prototype)
Edit (include right click)	-
Help documentation	√
Portability	-