CHAPTER 5 - SYSTEM CODING AND TESTING

5.1 OVERVIEW

Coding is the process that translates the detailed design representation of the software into a programming language. The translation process continues when a computer accepts source code as input and produces machine code. After system coding, system testing is carried out. An effective test design enables the developer to focus efforts where they are most clearly needed. If system testing is conducted thoroughly, it will uncover errors in the software. Indirectly, it will demonstrate the working systems capability and whether the system is working according to specifications and performance requirements (Pressman, 2001).

5.2 DEVELOPMENT TOOLS USED

In this project, the development tools used include:

i. Microsoft Visual Basic 6.0 (Perry, 1999)

The program modules and interface were developed using Microsoft Visual Basic 6.0. Visual Basic has the ability to combine text-based programming with visual techniques using icons and menus. In writing an application, the user interfaces were first developed by directly drawing on the screen using an editor. The visual interface provides the character for a program and interacts with the user. The programming language works behind the scene to connect the visual elements. The codes are then attached to the objects and will respond to specific events.

ii. Microsoft Access 2000

The database was developed using Microsoft Access, a relational database management system (DBMS) (Microsoft Corporation, 2002). Data controls are used

to perform database administrative tasks from the interfaces created using Visual Basic (Amundsen and Smith, 1996).

iii. Crystal Report

Crystal Report brings fourth the power of Windows reporting. Crystal Report allows the use of graphics, colours, underlining, and varied font capabilities (Crystal Report, 2001). The two reports generated by MPID are growth and growth rate of mobile phone usage and forecasted growth of mobile phone usage.

iv. Macromedia Flash (Ulrich, 2001)

Animation and audio were developed using Macromedia Flash. The software enables users to combine text, graphics, animation and audio into a multimedia presentation. The animation file was incorporated into the introduction section.

5.3 PROGRAMS DEVELOPMENT

During coding, the design of MPID was translated into a machine-readable format. The design of MPID was done meticulously. The code generation was accomplished in a detailed and mechanistic manner.

The MPID system consists of different types of objects: forms, controls and program modules. Each of these is elaborated further in the following sections.

5.3.1 Forms

In Visual Basic, a form is where the graphical user interface (GUI) is displayed. The GUI is the visual portion of the system that enables users to interact with it. This is where the users enter an input to the program and where the program displays its output. To customize

forms, the toolbox is used. The toolbox is a collection of tools that acts as a repository of controls that can be placed on a form (Perry, 1999). MPID has altogether seventeen forms, which are divided into three categories - General Forms, Data Entry Forms, and Report Forms.

General forms are mostly used as interfaces for interaction with users. They consist of:

i. frmMain

This is the main form used in MPID. It has a toolbar and a menu that enable users to navigate the system with ease through the system.

ii. frmData

This enables users to select the data group: university students, working group or both.

iii. frmAbout

This displays information pertaining to the version of MPID and Intellectual Property Rights (IPR).

iv. frmFactors

This displays all the features and the description of the features.

v. frmForecast

This enables users to make forecast on the growth of mobile phone usage in relation to the features selected by the user. The forecasted values are stored in a table and displayed graphically in a chart.

vi. frmRegenerateForecast

This form enables users to narrow down the range of year of the forecasted results from frmForecast.

vii. frmRegenerateForecastChart

This displays the results based on the range of years from frmRegenerateForecast in graphical formats (line chart, bar chart or 3D bar chart).

viii. frmCompare

This enables users to select the year and features to make a comparison of the forecasted results.

ix. frmCompareForecast

This displays the comparison of the forecasted results in a line chart, bar chart or 3D bar chart.

x. frmRegenerateForecastComp

This form enables users to narrow down the range of year of the forecasted results from frmCompareForecast.

xi. frmRegenerateForecastCompChart

This displays the results based on the range of years from frmRegenerateForecastComp in graphical formats (line chart, bar chart or 3D bar chart).

xii. frmIntro

This is the introductory screen that enables users to understand the flow of the system.

xiii. frmIntroFlash

This displays the short Flash introductory animation that gives users an overview of MPID.

xiv. frmMobileGrowthRate

This displays the number of subscribers and growth rate of mobile phone usage for the past twelve years. The data is displayed in chart form (line chart, bar chart or 3D bar chart).

xv. frmViewCat

This displays the data of the features by year. The data is displayed in chart form (line chart, bar chart or 3D bar chart).

Data Entry forms are used as a medium for users to insert, modify and delete data from the database (dbase.mdb). Data Entry forms include:

i. frmAddFactorValue

This allows users to add the total number of purchasing preferences of mobile phone by year. This data will be saved into the associated tables.

ii. frmAddFactors

This allows users to add an additional feature. A table would be created for every additional feature.

iii. frmAddGrowth

This allows users to add the number of mobile phone users.

iv. frmDeleteFactorValue

This allows users to delete the total number of purchasing preferences of mobile phone by year. This data will be deleted from the associated tables.

v. frmDeleteFactors

This allows users to delete the selected features. The associated feature tables will be deleted.

vi. frmDeleteGrowth

This allows users to delete the number of mobile phone subscribers.

vii. frmModFactorValue

This allows users to modify the total number of purchasing preferences of mobile phone by year.

viii. frmModifyFactors

This allows users to modify the selected features. The associated features table will be modified.

ix. frmModifyGrowth

This allows users to modify the number of subscribers.

Report Forms consist of the reports generated by Crystal Report. There are two report forms:

i. frmViewerGrowth

This displays the report on the total number of subscribers by year and the growth rate of mobile phone usage in a tabular and graphical form.

ii. frmViewerForecast

This displays the report on the forecasted growth of mobile phone usage in a tabular and graphical form.

5.3.2 Controls

Controls are reusable, predefined components used for visual programming. Controls serve as a building block that can be quickly combined to create a working application. Visual Basic has two types of controls, namely, intrinsic controls and ActiveX Controls. Intrinsic

controls are default controls provided in Visual Basic Toolbox (Deitel, Deitel and Neito, 1999). The list of intrinsic controls used in MPID includes CommandButton, Data, Frame, Image, Label, Line, Shape, TextBox, ComboBox and ListBox.

ActiveX controls must be loaded into a Visual Basic object by adding components from the control library. Generally, ActiveX controls are used in the same manner as intrinsic controls (Deitel, Deitel and Neito, 1999). ActiveX controls include SSTab, DBCombo, ShockwaveFlash, DataList, DataCombo, MsChart, CrytsalReport, MapiSession, CommonDialog and MsFlexGrid.

5.3.3 Program Modules

There are three types of program modules in MPID:

i. Form Modules

Form modules consist of smaller entities called procedures. There are four types of procedures: event procedures, Visual Basic procedures, sub-procedures and function procedures. Event procedures respond to events. Visual Basic procedures are provided by Microsoft to perform common tasks. Visual Basic allows programmers to create their own procedures called Sub-procedures and Function procedures to meet the unique requirements of the problem that they have to solve (Deitel, Deitel and Neito, 1999). There are altogether one hundred and forty-five form modules in MPID.

ii. Code Modules

Unlike form modules, code modules do not have Graphical User Interface (GUI) and contain codes only. Code modules are maintained in separate file names ending in .bas (Deitel, Deitel and Neito, 1999). There are twelve code modules in MPID.

iii. Class Modules

Similar to form modules, class modules have a general declaration. Unlike code modules, class modules have only two elements associated with them, namely, Initialise and Terminate. Class modules are important as they can easily be reused in the development of other Visual Basic projects (Deitel, Deitel and Neito, 1999). There are two class modules in MPID.

5.4 TESTING

There are many types of testing techniques such as functional testing, equivalence partitioning, boundary test, structural testing, stress test, and recovery test. Three types of testing were performed in the development of MPID: unit testing, module testing and integration testing.

5.4.1 Test Plan

Before the testing phase begins, a test plan must first be prepared to guide the testing process. The effort to build a test plan is an iterative process, requiring feedback and agreement between the various project participants on the defined approaches, test strategies and timeline for performances (Dustin, Rashka and Paul, 1999).

The purpose of the test plan can be summarized as follows (Dustin, Rashka and Paul, 1999):

- It provides guidance necessary to support the test programme.
- It establishes the nature and the extent of tests deemed necessary to achieve goals and objectives.
- It outlines an orderly schedule of events and activities that use resources efficiently.
- It provides assurance of the highest level of test coverage possible through the creation of requirement traceability.
- It outlines the detailed contents of test procedures scripts and describes how the test procedures scripts will be executed.

The test plan is a document that guides test execution through to conclusion, and it must be updated to reflect any changes. The outline of MPID test plan is shown in Appendix D.

5.4.2 Unit Testing

Unit testing is used to verify that software codes perform adequately and correctly implement the detailed design (Dustin, Rashka and Paul, 1999). Unit testing focuses on verification efforts for the smallest unit of the system design, which includes subroutines or functions. Unit testing is a part of the white box testing technique, which makes use of the control structure of the procedural design. Unit testing can be conducted in parallel for multiple modules. This testing is simplified when a module with high cohesion is designed such that a module addresses only one function and the number of errors can be more easily predicted and uncovered (Pressman, 2001). Errors documented as a result of unit testing

can include logic, overload or overflow of range, timing and memory leakage detection errors (Dustin, Rashka and Paul, 1999). Using this process, error detection for the coding and logical mistakes hidden in MPID can be easily detected.

5.4.3 Module Testing

Module testing is performed after completion of each system module. Module testing is needed to ensure that the module demonstrates and works according to the specifications of the system. In module testing, each module is treated as an independent component. In this case, the black box approach is used where the characteristics are evaluated through a study of the correlation between the inputs and the outputs in MPID.

5.4.4 Integration Testing

Integration testing is a systematic technique for constructing the program structure while conducting test to uncover errors associated with interfacing. The objective is to take unit tested modules and build a program structure that has been dictated by design. The following steps summarize the whole approach used during the integration testing:

- The content model for MPID is reviewed to uncover errors.
- The design model for MPID is reviewed to uncover navigation errors.
- The processing components are selected and MPID is unit tested.
- Integration tests are conducted on the architecture constructed.
- The MPID is tested for overall functionality and content delivery.
- MPID is implemented in two different environmental configurations, namely Windows 98 and Windows 2000, and is tested for compatibility with each configuration.

Based on the testing techniques, test cases were designed to discover the possible types of errors hidden in MPID. These test cases are included in Appendix D.

5.5 DOCUMENTATION

Documentation is the recording of all the activities planned and done during the development phases of the application. Different kinds of documents are produced along the development lifecycle. The documents produced in this research include project management documents, data collection questionnaire, regression results, requirements specification, test cases, evaluation questionnaire and MPID evaluation results. Project management documents include project schedule, project tracking, project reviews and project cost reports. The documents prepared for MPID are as follows:

1. Survey on the Preferences of Mobile Phone Users towards Mobile Phone Features (Appendix A) – document created to investigate the relationship between the advancement in S&T and growth of mobile phone.
2. Regression Results of the Mobile Phones Features and Number of Subscribers (Appendix B) – results generated by SPSS® for Windows version 11.0 based on one or two mobile phone features and the number of subscribers.
3. Requirements Specification (Appendix C) - describes the requirements specification for the MPID modules.
4. MPID Test Plan and Test Cases (Appendix D) – document steps in testing the functionality as specified in the design specification document and the requirements document. It also verifies whether the software delivers the interfaces correctly.
5. MPID System Evaluation Questionnaire (Appendix E) – questionnaire form distributed to users to gather feedback.

5.6 SUMMARY

This chapter describes the coding phase and the testing phase of MPID. The development tools and the program development were discussed. Three types of testing techniques used were also discussed. A test plan was produced to perform the test steps systematically.