# Chapter 4: The Development of A Risk Management Tool

## 4.1 Introduction to A Risk Management Tool: The Statistical Manager

To deal with the software development problems mentioned in Chapter 1, and to ensure cost-effectiveness, I believe that organizational based software risk management tool will be able to assist in solving most, if not all, of these problems. The tool that I have developed, called The Statistical Manager is a project management tool that emphasizes on software risk management. It is a windows-based software tool designed specifically for the actual management of risk in a project or program of an organization. It does not eliminate risks that plague a software project, but it will help the developers to forecast risks and make plans to accommodate the inevitable delays, cost overruns, and other unpleasant surprises.

Basically, there are two main parts in The Statistical Manager: The Database and the Statistic. The first allows the users to capture, maintain, query, and report internal software development key parameters easily and confidentially; whereas the second allows the users to perform some useful statistical functions like median, standard deviation, variance, coefficient of variance, and more importantly, correlation, regression and curve fitting.

One of the keys to successful project execution is a matured and pervasive organization database. Software development individuals can bring applicable materials to the effort of risk management, but the organization, as a whole may not have a reliable industrial database. This fact will significantly impact the effectiveness of the organization's risk management effort.

With The Statistical Manager, software development organizations can capture project key attributes such as process model, project phases, project cost, duration, project complexity, risk conditions and consequences, mitigation strategies and more. The resulting database, which is accumulated over time and many projects, establishes a strong organizational project foundation. This foundation can then be used to:

- Identify and manage future project risks based on lessons learned before they become unmanageable;
- Identify problem areas early and focus on the items that have deteriorated the previous projects;
- Compare projects so that organizations would gain a better understanding of what risks they had undertaken and how these risks had impacted their project as well as how these risks were being dealt with;
- Correlate project parameters of interest (like software size and risk level, project complexity and risk level), deriving a linear regression equation, generating a line fit forecast , testing on the significance of relationship, and using this relationship to predict relevant parameters of currently undertaken or future projects;
- Discover patterns and themes to help identify common risks and formulate generalizations about risks;
- View, generate and print reports to suit a particular organization's needs;
- Support planning of future projects and make decisions based on data as essential information becomes readily available to drive project decision-making ;
- Develop organizational internal rules of thumb.

A wise man once said, "Those who do not study the past are destined to repeat the mistakes that they have made" and "Those who do study the past will know what to expect in the future". Learning from mistakes is an important element of individual and organizational development. Accumulating knowledge of mistakes and risks involvement from the past and applying these experiences into future projects is a powerful strategy. The Statistical Manager works on the basis of this method. It uses statistical forecasting[X] to "study the past" and assists planning in the future. It allows reviews of organization learning about similar projects and the special circumstances that is likely to threaten the planned project. Actions can then be developed to ensure that these problem areas are adequately attended to, by proper utilization of manpower, resources or other relevant measures.

The Statistical Manager allows users (system developers) to easily spot trends by displaying historical data graphically. It is possible to quickly identify and predict what will happen on the project and to take appropriate measures. At the same time, it also provides meaningful management indicators.

The Statistical Manager uses powerful SQL driven statements that enables the users to explore and analyze their data. The users are able to query, retrieve, sort, view and report any data from a project or multiple projects instantly.

The Statistical Manager is equipped with a security system. The users are categorized into two levels: The superuser (administrator) and the normal user. The superusers are allowed to setup and remove normal user's accounts. They can create, edit, view, and

[X] Forecasting technique uses past trends in data to predict likely future trends. The techniques are statistical in nature and attempt to uncover meaningful patterns in historical data. The big assumption is that the future will behave like the past.

perform statistical analysis on all project data. Normal users are only allowed to create and edit own project data but are allowed to view all project data and perform statistical analysis on all the project data. The administrator can establish user permissions for the entire organization and project team, enabling full control over projects, while increasing team involvement and awareness of risk. User permission is password protected. Security features include a track record of all users who log on to the tool.

## 4.2 Objectives of the Tool

The main purpose of The Statistical Manager is to assist software project development personnel to capture and analyze various types of project data statistically. The results of analysis can be used to identify, predict and mitigate risks on new projects. Various meaningful conclusions and precise decisions can then be made in the various stages of the project development.

There are two main parts in The Statistical Manager: The Database and the Statistic. The Database deals with the gathering and retention of project and risk information and thereby aids every step of risk management. It provides standard data definitions and data format to ensure that data is systematically gathered, properly checked, accurately entered into the database, and effectively managed to allow reliable analysis. It also provides risk record archives, which makes tracking and analyzing of risk much simpler. Through the study of past records, software engineers can plan and reduce, if not eliminate, various software risks. Mitigation of these risks will also be more effective, because past methods and their results are readily available for evaluation. If no project shares risk information, the organization's body of knowledge won't exist. The goal of this database is to create that body of knowledge and make it available to the organization. They can then use this database to facilitate planning and improve decisions.

The Statistic deals with the subsequent manipulation of the collected data. It can help the users to resolve a number of key software engineering decisions, such as to answer questions like, " Is there a relatively high correlation between risk level and project size?", "How much fund should be invested in this project?", "How long this project could possibly take?". The users can ease the stress and feel confident that they are making the right choice when facing situations where they have to give the "go-ahead" decisions. New projects can review the organization's risk factors, as well as experience on methods of handling specific risks. The overall objectives of this tool are:

- To serve the software engineering community in the role of gathering, analyzing, and disseminating data and information on the impact of software risks as well as practices leading to improvement in the software product.
- To build a strong and valuable software project database by continually adding relevant data in the context of the organization; and hence enabling software organization to conduct its own risk assessment, quickly, easily and confidentially.
- To help the users, especially project managers to identify and manage the items of risk before a project commences. This is important because risk items might become more serious if appropriate action is not taken in advance.
- To let software development personnel to obtain information on common risks and mitigation strategies, thereby making informed decisions on the direction of software development programs.
- To discover patterns and themes that help identify common risks and formulate generalizations about risks;

- Manage risk items at an early stage of a project development. Risks can be identified and resolved/reduced as early as possible without propagating to the later stages at which time they might become unmanageable and finally affecting the overall development of the software.
- Allows comparison between projects so that organizations would gain a better understanding of what risks they had undertaken and how these risks had impacted their project as well as how these risks were being dealt with;
- Gives early warnings/alerts to software developers of possible risks and by so doing avoid rework, cost overrun and schedule delay.
- Provides a means to effectively communicate project information, particularly pertaining to risk information.
- To guide projects establish an initial baseline set of risks and mitigation plans – one of the key steps for putting risk management in place.
- Provides possible/potential solutions for planning and optimization of future project execution by showing a basic “roadmap” and insights, which will help to understand project status, and the actions that should be performed next.
- To increase the software risk awareness. It is hoped that with the introduction of this tool, software risk awareness will increase. Software organizations will have positive attitudes towards risk management and will take the initiative to manage risks.

## 4.3 Functional Requirements

The functional specifications define what a software system should do. The Statistical Manager should be able to perform the following:

- Capture organizational key project's parameter.

- Develop and maintain risk management database for software development organizations.
- Provide sufficient information in the risk management database to allow initial project resource planning as well as reasonably early "go/no go" decisions.
- Be based on the most commonly used software life-cycle phases and activities.
- Performs basic statistical and more advanced analysis.
- Correlate project parameters of interest (like software size and risk level, project complexity and risk level), derive a linear regression equation, generate a line fit forecast , test the significance of the relationship, and use this relationship to predict relevant parameters, with confidence limit, of currently undertaken projects or future projects;
- Help software developers make comparisons and identify risks which are relevant to their currently undertaken projects to past projects based on the results of analysis.
- Forecast risks that might be encountered before proceeding to development.
- Generate a list of watch items. These may include commonly occurring risks.
- Set criteria (s) to view specific project data.
- View, generate and print reports to suit individual needs.
- Back up the database.

## 4.4 Non-Functional/Performance Requirements

The performance specifications specify the quality that a software system should achieve. The Statistical Manager should achieve the following quality characteristics:

- ***Be easy to learn and use (usability).*** The tool should have simple and understandable user interfaces. It should include help menus and examples sufficient to answer users' doubts and provide training. The amount of formal training required to use the tool should be minimal. The required inputs from the users should be well defined, and visibility into internal equations and theories should be provided.

- ***Be reliable (reliability).*** The tool should be robust and should not fail/crash frequently. It should continue to operate correctly according to its intended functions with require precision, despite the introduction of invalid inputs.

- ***Be able to produce correct outputs (correctness).*** The tool should be free from design and coding defects. It should satisfy its specifications and fulfill the user's expectations. For instance, it should provide accurate statistical calculations and displays.

- ***Be maintainable (maintainability).*** The coding of the tool should be self-descriptive, properly structured and concise to attain ease of understanding, debugging, testing and maintaining as well as future modifications. The tool should be easily corrected if an error is encountered, adapted if its environment changes, or enhanced if the users desire a change in requirements. It should also have a proper documentation.

- ***Integrity.*** The tool should be able to withstand attacks (both accidental and intentional) on its security. This should include integrity to its components: program and data.

- ***Be cost-effective.*** Should be able to develop the tool with available resources. When in operation, it should not consume high-cost resources like additional computer hardware and peripheral devices.

- ***Reusability.*** The components/modules of the tool should be adaptable to (reuse) other applications with minimal changes.

## 4.5 Development Methodology

Many methodologies are available for the development of a software system. The method that I employed in developing this tool, the object-oriented, is currently gaining widespread popularity. I employed Object Modeling Technique [Rumbaugh et al., 1991] in the tool development. Apart from fulfilling one of the research project requirements and for personal development, the reason for choosing this method is that object-oriented paradigm reduces the need for testing. The software products are easier to maintain because their structures are inherently decoupled. This leads to fewer side effects when changes are made. In addition, object technologies allow reuse, which leads to object-oriented systems being easier to adapt and scale; i.e. large systems can be created by assembling reusable subsystems.

The Statistical Manager is developed with the use of Visual C++ running on an IBM compatible Intel Pentium PC under Windows 95 operating system. The Windows programming features are fully made use of in the development process of this tool. Appropriate software engineering concepts, principles, tools and practices are also employed to ensure that the product developed is of high quality, reliable and practical.

The Statistical Manager development is based on the following principles:

- Software organizations need a mature and pervasive organization database. Software development individuals can bring needed materials to the effort of risk management, but the organization, as a whole does not have a reliable industrial database. This fact significantly impacts the effectiveness of the organization's risk management effort.

- It is very difficult to make a good decision in the absence of any information about software projects. Such information can be of considerable economic value in deciding problems of uncertainty.
- Risks need to be identified when evaluating a project's proposal and not after the contract has been signed. A strong and reliable organizational database is the cornerstone for accurate identification and analysis of project risks. Organization will have a basis to reject the project proposal if the result of analysis is pessimistic.
- Once the project has commenced, risks and mitigation strategies need to be identified in the early phases of project development, since the major risk reduction leverage is in the early phases, and when costs are lower and potential impacts are less. Again, a reliable database is required.
- Risk identification and resolved strategies need to be planned early before they become obstacles to the successful project implementation. The irony is that the very optimism and enthusiasm a new project generates in its early stage can cause software developers to overlook clear signs of high-risk items, which later proves fatal. Many of the shortfalls, obstacles, and bottlenecks associated with software projects can be prevented or reduced, simply by first identifying the possible risks. A reliable database can assist in risk identification and mitigation.
- It is very difficult to convince the affected parties that a serious problem has developed, especially in the early stages of a project. A reliable set of data is needed to convince them. Again, individual organizations need to have their own set of data.
- Risks have different implications to different stakeholders. A powerful organizational database is required to provide support for dealing with these different stakeholders and their needs.

- The people actually executing a project know best about the project. When project parameters are retained, members can learn from each other, so that all in that organization will have a basis for continuous improvement. By looking into the risks common to all software projects with similar characteristics, team member will be able to "de-personalize" similar risks when there are identified, and can learn from what worked (and did not) when other projects mitigated similar risks.
- Project data, specifically risk factors need to be analyzed relative to the organization or in the context of the organization, and not by using somebody's data. The reasons are simple: every organization has its own ability to construct a software product and its unique management style. Someone else's data may not be relevant and may be very difficult to fit into use.
- Most project data are forecasts, and are most effectively analyzed based on reliable and proven methods such as statistical analysis.

## 4.6 Targeted Users

The Statistical Manager is a purpose-specific tool that serves to facilitate the software development process. Its targeted users are software developers especially project managers. These people are professionals who have computer literacy and are exposed to the computer environment very frequently, and are able to operate computers at they fingertips. Therefore, The Statistical Manager is designed to have more professional interfaces, without unnecessary graphics, pictures or sound. Besides computer literacy, the users need to have some statistics knowledge in order to fully understand this tool. However, to make the tool more understandable and user-friendly, the tool comes with a Tour Guide and a HELP assistant.

## 4.7 Software selections

Computer programming languages have undergone dramatic evolution. There are a number of programming languages available, which not only facilitate programming works but are also useful in many other engineering fields.

Today's programs incorporate sophisticated user friendly interfaces, involving multiple windows, menus, dialog boxes, and the myriad of metaphors which we are familiar with. Programs are becoming increasingly interactive. Object-oriented programming attempts to respond to these needs, providing techniques for managing projects of enormous complexity, achieving user-friendliness features and allows reuse of software components.

In this project, I am restricted to use only a number of object-oriented programming languages: C++, Visual C++, Visual J++ or Java. Visual C++ 6.0 was chosen to develop the Statistical Manager and runs on the Microsoft Window 98 operating system. The reasons being:

- Windows programming environments and object-oriented programming are increasingly important now. It will be an added advantage to master this type of programming skill.
- It allows development of application programs in a shorter time frame. Developers spend less time building applications, less time coding, less time compiling, and less time debugging, while enjoying greater component reuse.
- It is equipped with **IntelliSense Technology** that greatly simplifies coding with auto list members, parameter information, type information, code comments, and

complete word that eliminates the need to memorize complex syntax, parameters, and component properties.

- **Smallest ActiveX Controls** can be easily built in and allows creation of user-friendly and highly interactive interfaces.
- There is an **Edit & Continue** function in the debugger. Developers can edit code while debugging without having to quit the debugging session, rebuild, restart the debugger, and return the application to the state where the problem occurred.
- **Dynamic ClassView** allows easy navigation of code and save time as changes like adding a variable or method are reflected immediately in **ClassView**.
- The functionality of Active Documents from Microsoft Word, Microsoft Excel, and other applications can be easily and seamlessly integrated by the use of **Active Document Containment.**
- **Composite Controls** allows state-of-the-art ActiveX development and permit easy reuse of any ActiveX control.
- **Faster Compiler Throughput.** Compiler throughput on debug projects is as much as 30 percent faster, and on non-debug projects as much as 15 percent faster (without sacrificing any optimizations).
- Permits **MFC (Microsoft Foundation Classes) Applications** that run with better granularity and less code overhead in dynamic link scenarios.
- **Optimizing Compiler Keywords** that optimize with compiler performance keywords.
- **Delay Load Imports** feature that speeds up applications by deferring DLL loading until necessary for continuing execution.
- **OLE DB Consumer and Provider Templates** that simplify high-performance access to any data source: relational and non-relational, e-mail and file systems, text

and graphic, custom business objects, and so on-with new templates for building OLE DB consumers and providers.

- **Enterprise Visual Database Tool** allows viewing of tables, modifying data, and creating SOL-queries within the IDE for any ODBC- or OLE DB- compliant database.

- **Multiple Monitor Support** allows 'divides and conquers' concept. It allows running of an application on one screen and debug on another (Requires Windows 98 or Windows NT 5.0).

- **Visual Component Manager** that allows sharing of a wide range of component types and enabling component and code reuse.

- **ANSI/ISO C++ Conformance.** The Standard C++ Library conforms to the September 24, 1996 ANSI/ISO (X3J16) Working Paper. The bool, mutable, and explicit data types are now supported per the standard.

Nevertheless, other then the above-mentioned features, Visual C++ possesses certain deficiencies. To mention one, applications developed using this language are not portable; i.e. they are platform dependent. However, in the absence of these important features, Visual C++ is still a comparatively very powerful and useful programming language.

## 4.8 Hardware selections

An IBM compatible Intel Pentium Processor 166-MHz microcomputer (PC) equipped with a SVGA color monitor, installed with a Windows 98 operating system, a Visual C++

version 6.0 software and an ODBC[XI] 32-bit, a standard keyboard, a Microsoft mouse and a CD-ROM drive are used to develop Statistical Manager. A bubble-jet printer is also used for printing purpose. The reason for choosing a PC is because microcomputers are now popular and widely used due to the fast progress of the hardware technology that enable users to obtain a PC and its peripherals easily and at an affordable price.

For the users to run this tool satisfactorily, it is necessary to have the following minimum hardware and software requirements:

Hardware:

An IBM or its compatibles Intel Pentium 75-MHz Processor PC (Pentium 90-Mhz or higher microprocessor recommended)

Standard motherboard

16 MB RAM or above

Up to 30 MB of free hard disk space for full installation

A SVGA card, 600 x 800 resolutions, 16bit color monitor

A keyboard, a mouse or compatible pointing devices supported by Windows

A printer for printing of reports

Dual Speed CDROM for installation (Quad speed or higher recommended)

Software:

Microsoft Windows 95 or later.

ODBD 32 Bit.

[XI] Open Database Connectivity technology provides a common interface for accessing heterogeneous SQL databases. It is based on the Structured Query Language (SQL) as a standard for accessing data that provides maximum interoperability. A single application can access different SQL DBMS through a common set of code. This enables developing of application without targeting a specific DBMS. Database drivers are then added to link the application to the user's choice of DBMS.

## 4.9 Design of the Tool

The overall architecture of The Statistical Manager is depicted in Figure 4.1 and Figure 4.2. It consists of eight functional modules, six of them with sub-modules.

1. **Login Module**: This module is responsible for user's login. The user needs to provide a correct user name and a password (which is set in the **System Setting** Module) in order to be able to use this tool.

2. **Project Module**: This module has three sub-modules:
   - ♦ ***Create New Projects***: The user is required to type in the name of the project. A project ID and a risk ID will be automatically assigned and these IDs will be used as long as the project is present. A form which requires the user to fill in risk information and project data will pop up. All project data and risk information will be captured here. Various risk records may then be created for a project and different risk ID will be assigned to these risk records.
   - ♦ ***Open Existing Projects***: A user can edit project data and risk information here. He/she can choose to update (in this case, the existing one will be archived) or to overwrite the existing one.
   - ♦ ***Delete Projects***: A project that is not required can be removed from the database. However, for security reasons, only the owner of the project (the creator) is allowed to perform this deletion.

3. **Statistics Module**: This is the biggest module. Many statistical functions can be performed here. The order is as follows:
   - ♦ ***Select projects***: A user needs to select from an available list of projects which he/she is interested to perform the statistics.

## Software Risk Management Tool: The Statistical Manager Functional Decomposition Diagram

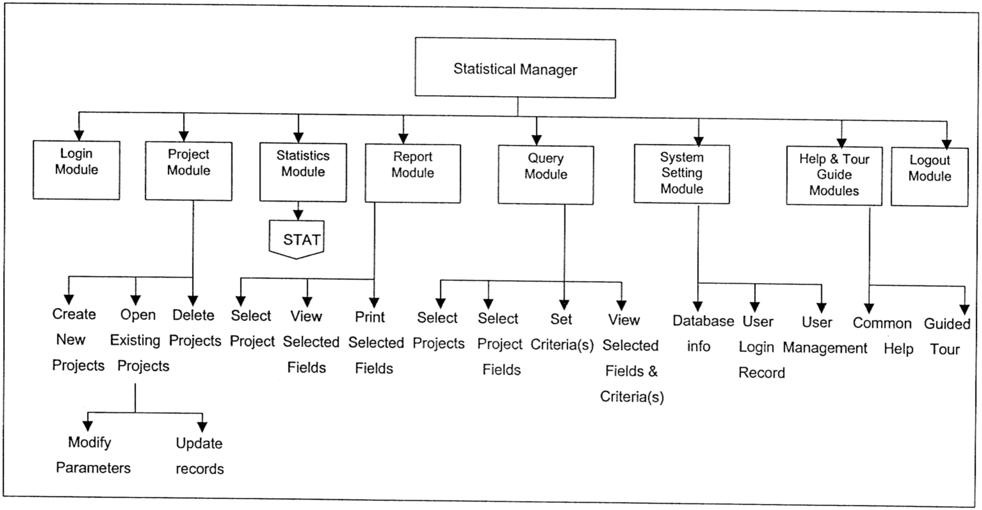


Figure 4.1: Functional Decomposition Diagram for The Statistical Manager

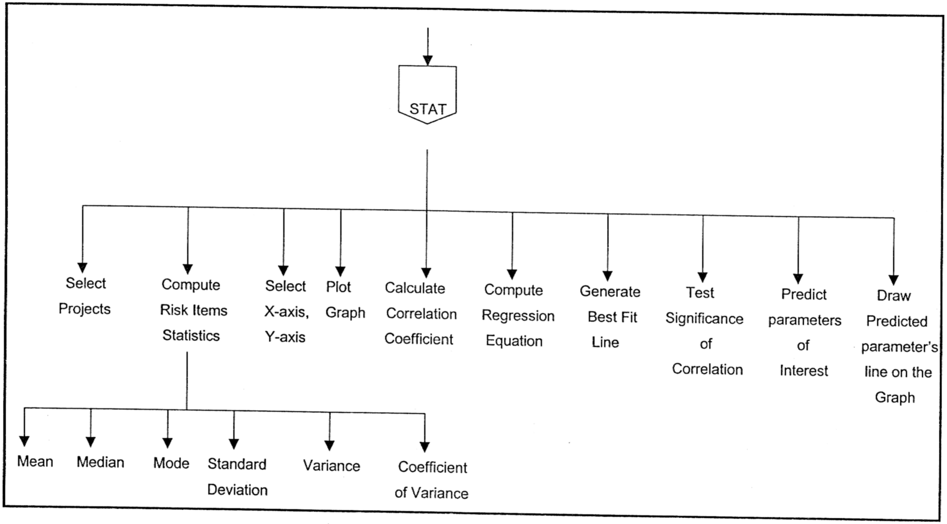

STAT
Select Projects
Compute Risk Items Statistics
Select X-axis, Y-axis
Plot Graph
Calculate Correlation Coefficient
Compute Regression Equation
Generate Best Fit Line
Test Significance of Correlation
Predict parameters of Interest
Draw Predicted parameter's line on the Graph
Mean
Median
Mode
Standard Deviation
Variance
Coefficient of Variance

**Figure 4.2: Functional Decomposition Diagram for Statistics Module**

- ***Select risk items***: A project can have many risk records. Again, a user will be prompted to select from an available list of risk records to be included in the calculation. Once this is done, the Mean, Median, Mode, Standard Deviation, Variance and Coefficient of Variance will be calculated and displayed.
- ***Select X-axis, Select Y-axis***: A user is required to select a pair of axis, one for X (independent variable) and the other for Y (dependent variable) to perform correlation and regression. When the user select X-axis, a list of projects with different project size measurements (function points (FP) and kilo lines of code (KLOC)) will be provided. The user will be asked to choose one of these measurements.
- ***Plotting graph***: Points of interested project's parameters will be plotted in graph display area on the screen.
- ***Calculate correlation coefficient***: This is for the user to find out the strength of correlation between the two chosen parameters. The value and comments will be displayed in the message box. If there is no correlation between these parameters, the next three functions will not be allowed to perform.
- ***Compute regression equation and generate best-fit regression line***: If correlation between the parameters exists, a regression equation in the form of Y = mX + c will be computed and shown in the message box. At the same time, a regression line will be drawn in the graph.
- ***Testing on significance of correlation***: This function is to find out the significance of the correlation, which is set at 95% level.
- ***Predict parameters of interest and draw predicted parameter's line on the graph***: When there is correlation between the parameters, the user can proceed to predict the parameter of interest, i.e. the dependent variable, Y. The predicted

values will be displayed in the message box. The relevant line will also be drawn in the graph.

4. **Report Module**: The user will be asked to select projects to be included in the reports. He/she can choose two formats of report: vertical listing by project or columnar listing by project's field. The user will be able to print out this report.
5. **Query Module**: The user can perform a variety of queries base on his/her needs. The name of the query needs to be entered. The user can specify which fields/parameters are to be included, set query criteria(s) and to view in ascending, descending or database orders with respect to the field specified.
6. **System Setting Module**: Database Information, Database Backup, Log File, User Management and User Login records are displayed here. The system administrator creates each user's account by specifying user's name, password, job title and type of user.
7. **Help Module**: The module provides two types of assistance: The context sensitive HELP file and the Tour Guide. The introduction to the tool, all doubts and questions from users can be found here. Tour Guide provides a step-by-step guidance to the users, from setting up of DBMS to detailed explanation on the use of the tool.
8. **Logout Module**: Responsible for the user log out from the tool.

Figure 4.3 and Figure 4.4 shows the dynamic model and functional model of The Statistical Manager respectively.

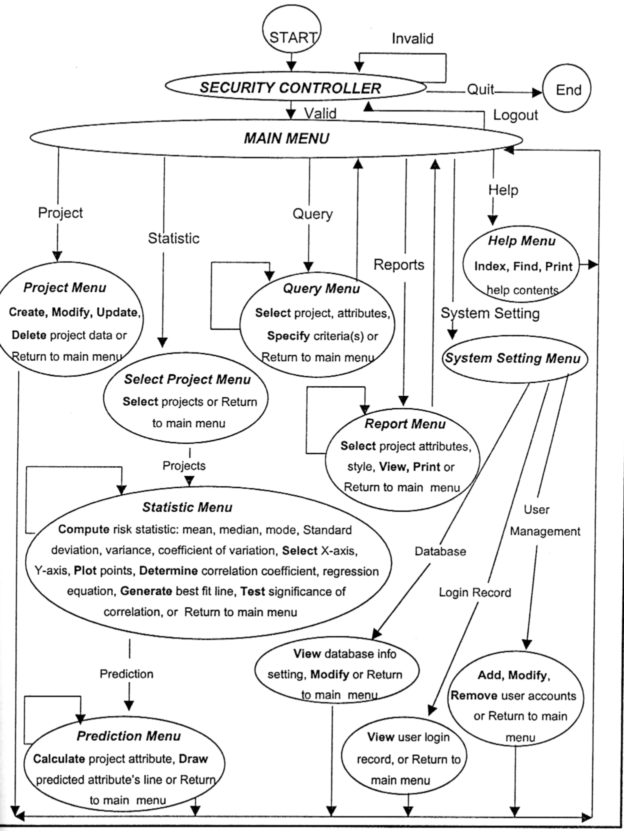

START
Invalid
SECURITY CONTROLLER
Quit
End
Valid
Logout
MAIN MENU
Project
Statistic
Query
Reports
Help
System Setting
Help Menu
Index, Find, Print help contents
Project Menu
Create, Modify, Update, Delete project data or Return to main menu
Query Menu
Select project, attributes, Specify criteria(s) or Return to main menu
System Setting Menu
Select Project Menu
Select projects or Return to main menu
Report Menu
Select project attributes, style, View, Print or Return to main menu
Projects
Statistic Menu
Compute risk statistic: mean, median, mode, Standard deviation, variance, coefficient of variation, Select X-axis, Y-axis, Plot points, Determine correlation coefficient, regression equation, Generate best fit line, Test significance of correlation, or Return to main menu
Database
User Management
Login Record
Prediction
View database info setting, Modify or Return to main menu
Add, Modify, Remove user accounts or Return to main menu
Prediction Menu
Calculate project attribute, Draw predicted attribute's line or Return to main menu
View user login record, or Return to main menu

**Figure 4.3: Dynamic Model(OMT) for the Statistical Manager**

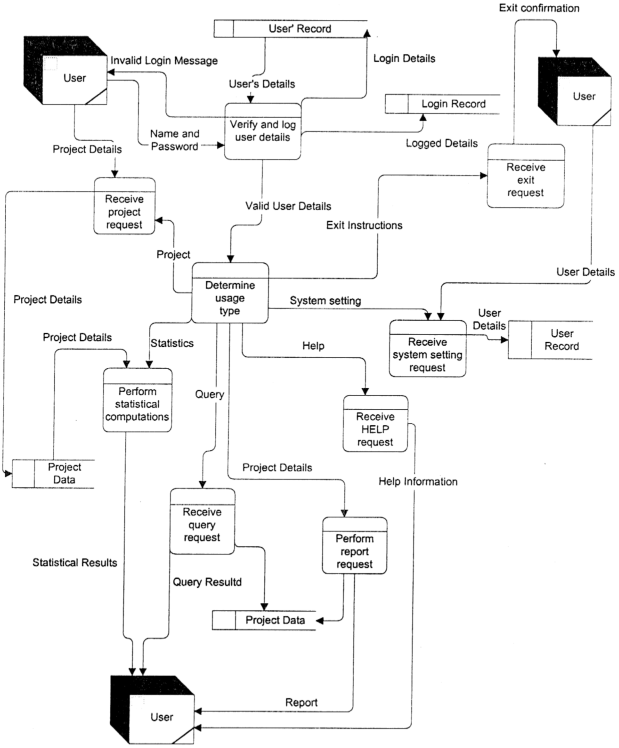
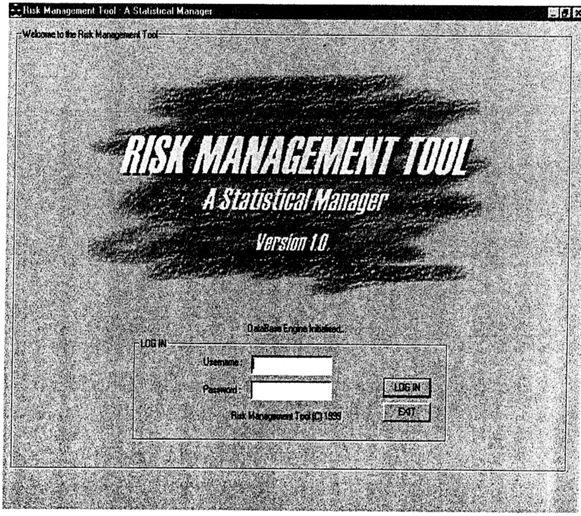


*Figure 4.4: Functional Model of The Statistical Manager*

There are 25 classes for The Statistical Manager. They are arranged in ascending alphabetical order with respect to the class name. Detailed design of each class is shown in Appendix B.

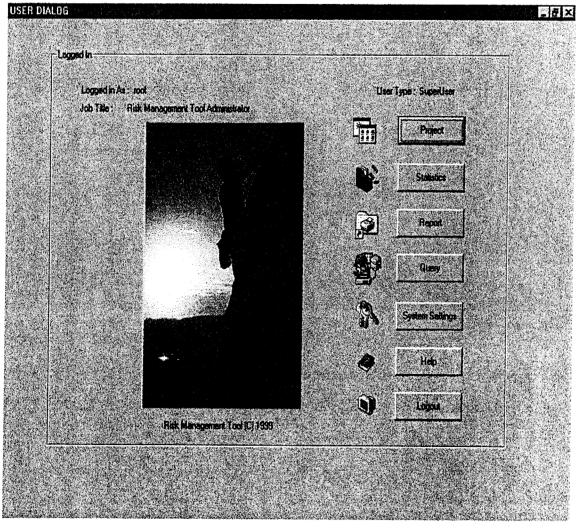
## 4.10 Screen Design of Statistical Manager

The following are some screenshots. The usage of the tool is illustrated further in Appendix C. These screenshots are not in their actual sizes. They are scaled down so that they can fit into these pages.


Risk Management Tool : A Statistical Manager
Welcome to the Risk Management Tool
RISK MANAGEMENT TOOL
A Statistical Manager
Version 1.0.
DataBase Engine Initialised.
LOG IN
Username :
Password :
LOG IN
EXIT
Risk Management Tool (C) 1999

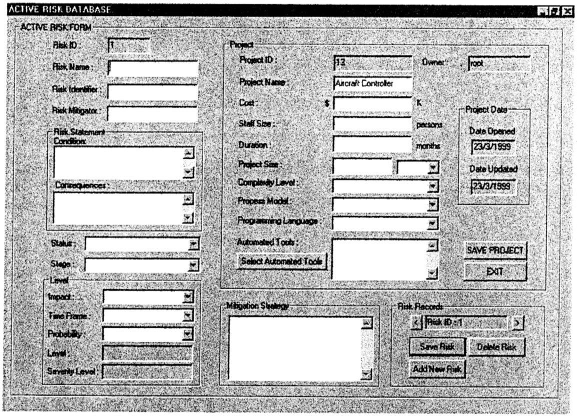
**Screen 4.1: Login interface.**

Screen 4.1 shows the start-up/opening screen. It also serves as a login interface. The user will be able to see this screen if a database source has been set up correctly. The name and version of the tool are displayed here. A username and a password of up to 10 characters long consisting of no special character must be entered correctly in order to log in to the tool. The System Administrator (Superuser) sets up user accounts in the **System Setting**. The user can exit from the tool by either clicking the **EXIT** button or the **CLOSE (x)** button at the upper right corner.



**Screen 4.2: The main menu screen.**

Screen 4.2 shows the main menu screen which will be displayed if a user has successfully logged in. The upper portion of the screen displays the user login name, the job title and the user type/rank. The main menu consists of seven command buttons, each with its specific functions, which will be explained in the later part of this chapter. Unique and meaningful names are given to each of this buttons to indicate its functions.


ACTIVE RISK DATABASE

ACTIVE RISK FORM

Risk ID: 1

Risk Name:

Risk Identifier:

Risk Mitigator:

Risk Statement

Condition:

Consequences:

Status:

Stage:

Level

Impact:

Time Frame:

Probability:

Level:

Severity Level:

Project

Project ID: 13

Owner: root

Project Name: Aircraft Controller

Cost: $ K

Staff Size: persons

Duration: months

Project Size:

Complexity Level:

Process Model:

Programming Language:

Automated Tools:

Project Date

Date Opened: 23/3/1999

Date Updated: 23/3/1999

Mitigation Strategy

Risk Records

Risk ID: 1

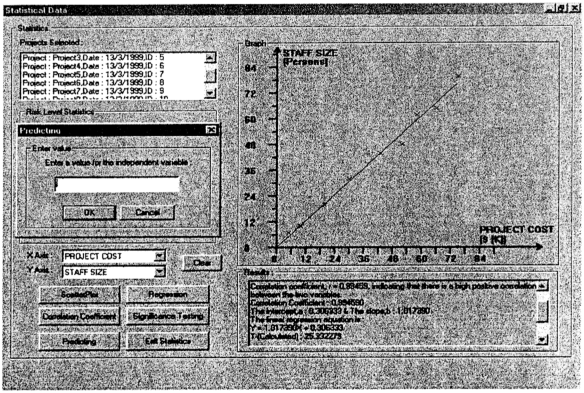
**Screen 4.3: User Input Form for Project data and Risk Information.**

Screen 4.3 will open when a user chooses to create a new project. It is the only user input form in the tool. All project data and risk information will be captured on this screen and saved into the database file. The form is separated into two sections, one section to capture risk information and the other for project data.

**Project ID** and **Risk ID** fields are automatically generated. **Project Date** and **Date Updated** are system dates. The owner (login name) and the name of the project (entered by the user) are also captured here.

All fields, except the **Automated Tools** field in this form must be completed before a user can save it. This is to ensure a systematic collection of project data and risk information. To enter data, simply click the intended field and type in the data. To maneuver between the fields simply hit TAB key as we would normally do to move around a window.

Many risks can exist for one project. A new risk record can be added by clicking the **Add New Risk** button. The risk number (automatically assigned) for a particular project is indicated under the **Risk Records**. The user can browse through these risk records, if there is more than one, by clicking the right or left direction arrows. **Risk Level** and **Severity Level** fields are derived fields. They are computed based on **Impact**, **Time Frame** and **Probability**. A complete list of definitions for all fields is provided in Appendix A.



```
Project : Project3,Date : 13/3/1999,ID : 5
Project : Project4,Date : 13/3/1999,ID : 6
Project : Project5,Date : 13/3/1999,ID : 7
Project : Project6,Date : 13/3/1999,ID : 8
Project : Project7,Date : 13/3/1999,ID : 9
```

X Axis : PROJECT COST

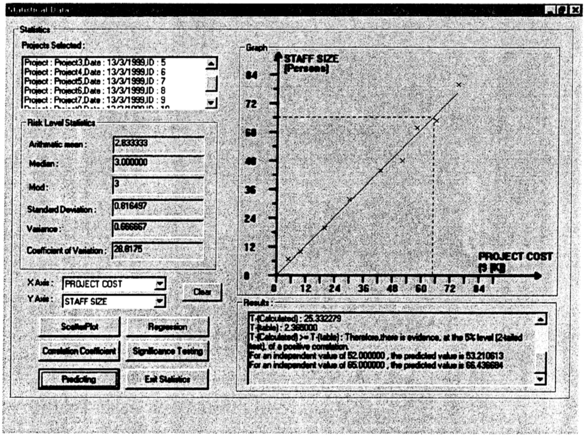
Y Axis : STAFF SIZE

```
Correlation coefficient, r = 0.89459, indicating that there is a high positive correlation
between the two variables.
Correlation Coefficient : 0.894590
The intercept,a : 0.306333 & The slope,b : 1.017390
The linear regression equation is :
Y = 1.017390X + 0.306333
T-(Calculated) : 25.332273
```

**Screen 4.4: Advanced statistical functions which shows results of correlation and regression.**

When a pair of X-axis and Y-axis is selected, a user can proceed to perform more advanced statistics. Coordinates of X and Y will be plotted on the graph area when **ScatterPlot** button is pressed. The strength of relationship between the two variables, r, will be calculated and displayed in the text box at the lower right corner when **Correlation Coefficient** button is clicked. If there are some relationships, **Regression** button can be clicked. A regression equation will be computed. This line will be shown as a red line in the graph.

The tool is programmed to 95% level (two-tailed test) to test the significance of the relationship. Results of the test will be summarized in the text box. For prediction, once the **Predicting** button is pressed, a dialogue box will appear, as shown above, for the user to enter the value of the independent variable.



```
Project : Project3,Date : 13/3/1999,ID : 5
Project : Project4,Date : 13/3/1999,ID : 6
Project : Project5,Date : 13/3/1999,ID : 7
Project : Project6,Date : 13/3/1999,ID : 8
Project : Project7,Date : 13/3/1999,ID : 9
```

| | |
|---|---|
| Arithmetic mean : | 2.833333 |
| Median : | 3.000000 |
| Mod : | 3 |
| Standard Deviation : | 0.816497 |
| Variance : | 0.666667 |
| Coefficient of Variation : | 28.8175 |

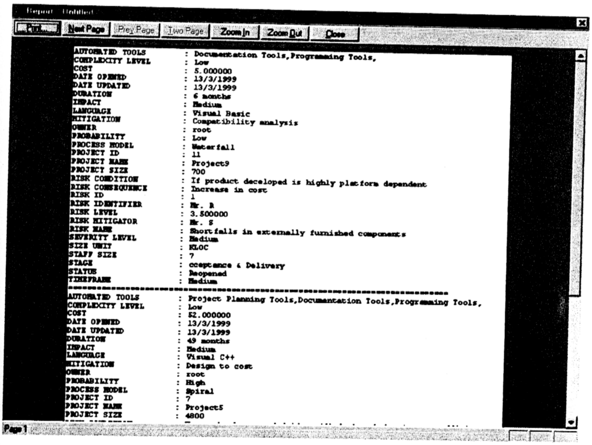
X Axis : PROJECT COST
Y Axis : STAFF SIZE

```
T-(Calculated) : 25.332279
T-(table) : 2.366000
T-(Calculated) >= T-(table) : Therefore,there is evidence, at the 5% level (2-tailed
test), of a positive correlation.
For an independent value of 52.000000 , the predicted value is 53.210613
For an independent value of 65.000000 , the predicted value is 66.436684
```

**Screen 4.5: Advanced statistical functions with prediction.**

When a user enters the independent X-value for prediction, the corresponding dependent value (Y-value) will be computed. The predicted value will be displayed in the text box; and a line (the green one as in above) will be drawn. Many predictions can be done and all the predicted values will be displayed to ease comparison. For the above example, if a project costs at $52K, the estimated staff size will be 53 persons; whereas for a project of $65K, the corresponding estimated staff size will be 66 persons.

The text box at the lower right hand corner displays all outputs of the statistical operations. To view all, press the scroll up and down at the right hand side of the text box. Examples of these outputs are provided in Appendix C.

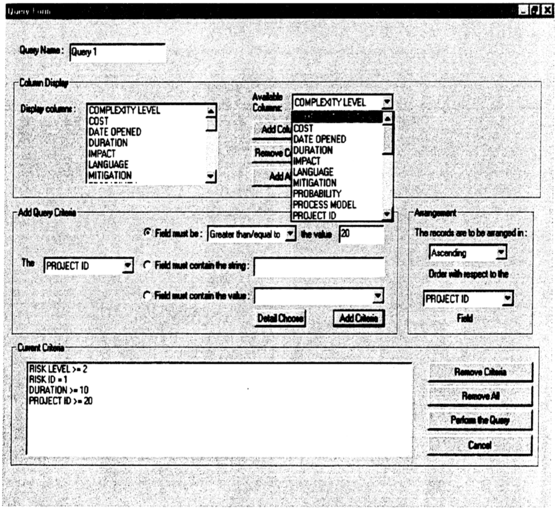


```
AUTOMATED TOOLS        : Documentation Tools,Programming Tools,
COMPLEXITY LEVEL       : Low
COST                   : 5.000000
DATE OPENED            : 13/3/1999
DATE UPDATED           : 13/3/1999
DURATION               : 6 months
IMPACT                 : Medium
LANGUAGE               : Visual Basic
MITIGATION             : Compatibility analysis
OWNER                  : root
PROBABILITY            : Low
PROCESS MODEL          : Waterfall
PROJECT ID             : 11
PROJECT NAME           : Project9
PROJECT SIZE           : 700
RISK CONDITION         : If product deceloped is highly platform dependent
RISK CONSEQUENCE       : Increase in cost
RISK ID                : 1
RISK IDENTIFIER        : Mr. R
RISK LEVEL             : 3.500000
RISK MITIGATOR         : Mr. S
RISK NAME              : Shortfalls in externally furnished components
SEVERITY LEVEL         : Medium
SIZE UNIT              : KLOC
STAFF SIZE             : 7
STAGE                  : cceptance & Delivery
STATUS                 : Reopened
TIMEFRAME              : Medium
==================================================================
AUTOMATED TOOLS        : Project Planning Tools,Documentation Tools,Programming Tools,
COMPLEXITY LEVEL       : Low
COST                   : 52.000000
DATE OPENED            : 13/3/1999
DATE UPDATED           : 13/3/1999
DURATION               : 49 months
IMPACT                 : Medium
LANGUAGE               : Visual C++
MITIGATION             : Design to cost
OWNER                  : root
PROBABILITY            : High
PROCESS MODEL          : Spiral
PROJECT ID             : 7
PROJECT NAME           : Project5
PROJECT SIZE           : 4800
```

**Screen 4.6: A sample report in the vertical listing format.**

Screen 4.6 is a sample vertical-listing format report in **Print Preview**. The report will exactly look like the above when printed. At Print Preview, the user has the flexibility to zoom in, zoom out, to see next page (if there is), to scroll up and down and to print. The page number is indicated at the lower left corner. To exit the preview window, click the **Close** or the close box (black X) in the upper right-hand corner of the window. If the user is not happy with the format as seen at the preview, he/she can choose **Edit** and alter the format manually. A name must be entered if a report is to be saved to a file.

The user can also perform copy/cut and paste to other applications.


Query Form

Query Name: Query 1

Column Display

Display columns: COMPLEXITY LEVEL, COST, DATE OPENED, DURATION, IMPACT, LANGUAGE, MITIGATION

Available Columns: COMPLEXITY LEVEL (COST, DATE OPENED, DURATION, IMPACT, LANGUAGE, MITIGATION, PROBABILITY, PROCESS MODEL, PROJECT ID)

Add Query Criteria

The PROJECT ID

[✓ Field must be: Greater than/equal to the value 20]

Field must contain the string:

Field must contain the value:

Arrangement

The records are to be arranged in: Ascending

Order with respect to the PROJECT ID Field

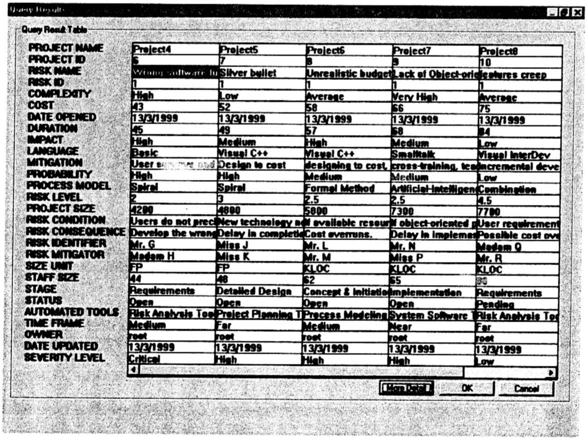
Current Criteria

RISK LEVEL >= 2
RISK ID = 1
DURATION >= 10
PROJECT ID >= 20

**Screen 4.7: Query setting dialog.**

In the query module, a query name is required from user. The user has the flexibility to choose project fields to display in the query result. To make a more specific query, the user can add query criteria(s), i.e. the conditions for the query search. Depending on the selection of the field, a corresponding value has to be entered. In the above example, the **PROJECT ID** field corresponds to a numeric field, so the tool will only accept numeric numbers. For a criterion to be taken into account, **Add Criteria** button must be clicked once for each criteria to be added. To remove criteria(s), just highlight and click the **Remove Criteria** or **Remove All** buttons.

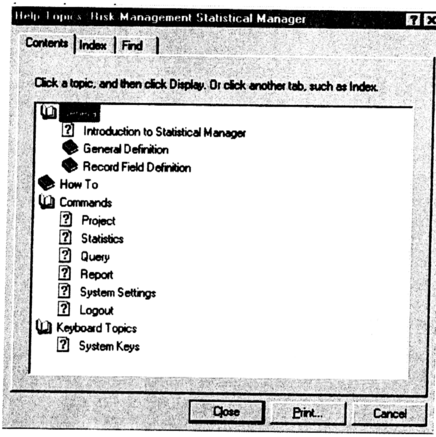
The user may request to arrange records in a specific manner: ascending, descending or not sorted (as in database order), with respect to a key (specified by the user). When everything is set, click the **Perform the Query** button to perform query.



| | | | | | |
|---|---|---|---|---|---|
| PROJECT NAME | Project4 | Project5 | Project6 | Project7 | Project8 |
| PROJECT ID | 6 | 7 | 8 | 9 | 10 |
| RISK NAME | Wrong software fu | Silver bullet | Unrealistic budget | Lack of Object-orie | features creep |
| RISK ID | 1 | 1 | 1 | 1 | 1 |
| COMPLEXITY | High | Low | Average | Very High | Average |
| COST | 43 | 52 | 58 | 66 | 75 |
| DATE OPENED | 13/3/1999 | 13/3/1999 | 13/3/1999 | 13/3/1999 | 13/3/1999 |
| DURATION | 45 | 49 | 57 | 68 | 84 |
| IMPACT | High | Medium | High | Medium | Low |
| LANGUAGE | Basic | Visual C++ | Visual C++ | Smalltalk | Visual InterDev |
| MITIGATION | User surveys and | Design to cost | designing to cost, | cross-training, tea | incremental deve |
| PROBABILITY | High | High | Medium | Medium | Low |
| PROCESS MODEL | Spiral | Spiral | Formal Method | Artificial-Intelligen | Combination |
| RISK LEVEL | 2 | 3 | 2.5 | 2.5 | 4.5 |
| PROJECT SIZE | 4200 | 4800 | 5800 | 7300 | 7700 |
| RISK CONDITION | Users do not preci | New technology n | il available resour | f object-oriented p | User requirement |
| RISK CONSEQUENCE | Develop the wrong | Delay in completio | Cost overruns. | Delay in implemen | Possible cost ove |
| RISK IDENTIFIER | Mr. G | Miss J | Mr. L | Mr. N | Madam O |
| RISK MITIGATOR | Madam H | Miss K | Mr. M | Miss P | Mr. R |
| SIZE UNIT | FP | FP | KLOC | KLOC | KLOC |
| STAFF SIZE | 44 | 48 | 62 | 65 | 80 |
| STAGE | Requirements | Detailed Design | Concept & Initiatio | Implementation | Requirements |
| STATUS | Open | Open | Open | Open | Pending |
| AUTOMATED TOOLS | Risk Analysis Too | Project Planning T | Process Modeling | System Software T | Risk Analysis Too |
| TIME FRAME | Medium | Far | Medium | Near | Far |
| OWNER | root | root | root | root | root |
| DATE UPDATED | 13/3/1999 | 13/3/1999 | 13/3/1999 | 13/3/1999 | 13/3/1999 |
| SEVERITY LEVEL | Critical | High | High | High | Low |

**Screen 4.8: A sample query**

Screen 4.8 shows a query example, sorted in an ascending order of Project ID field, and with a criteria cost > $40K. The user may view the cell items fully by clicking the **More Detail** button.



**Screen 4.9: Extract of the Help module.**

A complete **HELP** facility comes with the tool. To get assistance, what a user needs to do is to click the **HELP** button in the main menu, or to press F1 at any time when the tool is active.

**HELP** provides complete and comprehensive information of the tool. As shown above, **GENERAL** information introduces the tool to the user, and defines all terms and record fields used in the tool. **HOW TO** provides explanation on how to create a project, set up user accounts, perform query, print reports and so on. **COMMANDS** provides explanations on functions of each of the buttons in the main menu.

**KEYBOARD TOPICS** informs the user on the available keyboard and shortcut keys that the user can use.

## 4.11 Testing Strategies

In this project, object-oriented testing strategies are employed to test the tool. Three types of testing are carried out. They are Class Testing, Desk Checking and Validation Testing. Class testing is equivalent to unit testing for conventional software; but it is driven by the operation encapsulated by the class and the state behavior of the class. Such testing is carried out during tool development. In desk checking, each instruction is executed manually using test cases. Statistical computations are calculated manually, results are then compared to outputs from program to ensure that there is no error in formulations. Validation testing focuses on user visible and recognizable outputs from the tool. Methods from conventional testing which are appropriate for object-oriented software are used. They are black box (data-driven) testing which consists of equivalence partitioning, boundary value analysis and functional testing. Graphical user interfaces testing and help facilities testing are also conducted.

### 4.11.1 Black Box Testing

Black box testing is applied during later stages of the software development. It focuses on the functional requirement of the software. The purpose of the testing is to search for errors in the following categories:

1. Incorrect or missing functions;
2. Interface errors;
3. Errors in data structures or external data base access;
4. Performance errors;

5. Initialization and termination errors.

Black box testing requires a set of test cases. Test cases are developed and chosen in such a way so as to maximize the chances of detecting a fault while minimizing the chances of wasting a test case by having the same fault detected by more than one test case. The techniques used are equivalence partitioning combined with boundary value analysis and functional testing. Test cases for equivalence partitioning represent a set of valid or invalid states for input conditions, whereas boundary value analysis exercises the bounding values.

The followings show test cases selections for equivalence partitioning and boundary value analysis:

- For each range ($R_1$, $R_2$) listed in the input or output specifications, five test cases are used, corresponding to values less than $R_1$, equal to $R_1$, greater $R_1$ but less than $R_2$, and greater than $R_2$.
- For an input that requires a specific value, one valid (the specified value) and two invalid values (anything else) are defined.
- For an input that specifies a member of a certain set, one valid (a member of the specified set) and one invalid (a nonmember of the set) values are used.
- For input condition that is Boolean, one valid and one invalid values are defined.

The following black box test cases are created for The Statistical Manager.

Equivalence partitioning and boundary value analysis:

| Item | Test data/Testing | Results |
|---|---|---|
| Username, Password, | 1. <10 alphanumeric<br>2. 10 alphanumeric<br>3. > 10 alphanumeric | Accepted<br>Accepted<br>Rejected (too many) |
| JobTitle | 1. < 50 alphanumeric<br>2. 50 alphanumeric<br>3. > 50 alphanumeric | Accepted<br>Accepted<br>Rejected (too many) |
| DatabaseLoginName,<br>DatabasePassword | 1. < 255 alphanumeric<br>2. 255 alphanumeric<br>3. > 255 alphanumeric | Accepted<br>Accepted<br>Rejected (too many) |
| ProjectName,<br>RiskName,<br>QueryName,<br>RiskIdentifier,<br>RiskMitigator,<br>RiskStatement1,<br>RiskStatement2,<br>ProcessModel,<br>Language, | 1. < 100 alphanumeric<br>2. 100 alphanumeric<br>3. > 100 alphanumeric | Accepted<br>Accepted<br>Rejected (too many) |
| Mitigation | 1. < 200 alphanumeric<br>2. 200 alphanumeric<br>3. > 200 alphanumeric | Accepted<br>Accepted<br>Rejected (too many) |

| Item | Test data/Testing | Results |
|---|---|---|
| ProjectCost, TheValue (with respect to cost) | 1. Character instead of digit | Rejected (not a number) |
| | 2. < 0.00 | Rejected (not a positive number) |
| | 3. 0.00 | Accepted |
| | 4. 0.01 | Accepted |
| | 5. Between 0.01 and 999999999999998 | Accepted |
| | 6. 999999999999998 | Accepted |
| | 7. 999999999999999 | Accepted |
| | 8. >999999999999999 | Rejected (too many) |
| StaffSize, Duration, ProjectSize, TheValue (with respect to Staffsize, Duration, and ProjectSize). | 1. Character instead of digit | Rejected (not a number) |
| | 2. < 0 | Rejected (not a positive number) |
| | 3. 0 | Accepted |
| | 4. 1 | Accepted |
| | 5. Between 1 and 999999999999998 | Accepted |
| | 6. 999999999999998 | Accepted |
| | 7. 999999999999999 | Accepted |
| | 8. > 999999999999999 | Rejected (too many) |
| | 9. Any number with a decimal point | Rejected (not an integer) |

| Item | Test data/Testing | Results |
|---|---|---|
| RiskID, ProjectID, Owner, ProjectDate, Level, SeverityLevel, Mean, Median, Mode, StandardDeviation, Variance, CoefficientOfVariance | 1. Attempt to type, modify and delete | Rejected (Disabled) |
| Status, Stage, Impact, TimeFrame, Probability, ProjectSizeUnit, ComplexityLevel, AutomatedTools, X-Axis, Y-Axis, Arrangeement, Order | 1. Select from the available lists | Accepted |
| | 2. Attempt to type | Rejected (Disabled) |
| | 3. Attempt to delete and modify | Rejected (Disabled) |

## 4.11.2 Functional Testing

In functional testing, test data are based on the functionality of the module. Each item of functionality or function implemented in the module is identified [38]. After determining all the functions of the module, test data is introduced to test each function separately. The main functions which have been tested are:

1. Add a risk record
2. Back up a file
3. Calculate coefficient of variance
4. Calculate level
5. Calculate mean
6. Calculate median
7. Calculate mode
8. Calculate severity level
9. Calculate standard deviation
10. Calculate variance
11. Change password
12. Compose mitigation strategy

13. Compose risk statement condition
14. Compose risk statement consequences
15. Compute correlation coefficient
16. Compute regression equation
17. Compute t-value
18. Create a new project
19. Create a superuser
20. Create a user
21. Delete a project
22. Delete a risk record
23. Delete a superuser
24. Delete a user
25. Display messages
26. Display project particulars
27. Draw graph
28. Draw regression lines
29. Generate ID
30. Get estimate
31. Modify a project
32. Modify a risk record
33. Modify a user
34. Open an existing project
35. Open an existing risk record
36. Perform query
37. Perform sorting
38. Plot scatter points
39. Print report
40. Remove criteria(s)
41. Save a project
42. Save a risk record
43. Select field(s)
44. Select a project
45. Set database
46. Set criteria
47. Set displayed item
48. Set lines
49. Set registry
50. Set values
51. Set view order
52. Testing significance of correlation
53. Update a project
54. View project(s)
55. View risk record

In addition to these direct tests, the following additional tests have also been performed. In all testing, an appropriate error messages pop up for each attempt.

56. Attempt to backup a file with a name similar to the existing one.
57. Attempt to create a new project that is already present.
58. Attempt to create a new project without providing a name to the project.
59. Attempt to create a new risk record that is already present.
60. Attempt to create a report without specifying which projects to be included.
61. Attempt to create a superuser login name similar to the existing one.
62. Attempt to create a user login name similar to the existing one.
63. Attempt to delete a project that is not present.
64. Attempt to delete a risk record that is not present.
65. Attempt to delete a superuser when there is only one superuser in the system.
66. Attempt to open a project without providing a project name.
67. Attempt to perform prediction when correlation coefficient has not been calculated.
68. Attempt to perform query without providing a query name.
69. Attempt to perform regression when correlation coefficient has not been calculated.
70. Attempt to perform significance testing when correlation coefficient has not been calculated.
71. Attempt to perform statistics without selecting specified number of projects,
72. Attempt to plot a graph before a pair of X-axis and Y-axis is chosen.
73. Attempt to save a project when the input form is not completely filled.
74. Attempt to save a report with a name similar to the existing one.
75. Attempt to save an updated copy of project with a name similar to the existing one.
76. Attempt to set a criterion without specifying the reference field.
77. Attempt to set a database when the database driver is not present.

### 4.11.3 Graphical User Interfaces Testing

The GUIs designs have the same 'look and feel' throughout the tool. The following are the testing [84] and their respective results which have performed on the user interfaces:

**For windows:**

- Will the window open properly with respect to menu-based commands?

  ***Result: All windows opened properly when menu-based commands are activated.***

- Can the window be resized and scrolled?

  ***Result: Appropriate windows can be resized and scrolled.***

- Are all data contents contained within the window properly addressable with a mouse, function keys, directional arrows and keyboard?

  ***Result: All data contents within the window are addressable with a mouse. Appropriate functions and contents are reachable by function keys, directional arrows and keyboard, such as F1, Tab, Backspace, Esc, Enter, Del keys and others.***

- Does the window regenerate properly when it is overwritten and subsequently recalled?

  ***Result: When a window is overwritten by another program, it is properly regenerated when called again.***

- Are all functions that relate to the window available when needed?

  ***Result: All functions are visually displayed in the window. They are immediately available for use when a window is opened.***

- Are all functions that relate to the window operational?

  ***Result: All functions relevant to the window work very well.***

- Are all relevant pull-down menus, scroll bars, dialog boxes, buttons, icons, and other controls available and properly displayed for the window?

  ***Result: All items relevant to the window are displayed, or displayed when called. They are ready for use when displayed in the window.***

- When multiple windows are displayed, are the names of the windows properly represented?

  ***Result: Each window has a unique name to describe itself when multiple windows are opened.***

- Is the active window properly highlighted?

  ***Result: An active window is highlighted at title bar with Microsoft window default color, i.e. blue whereas non-active windows are grayed-out.***

- Do incorrect mouse picks within the window cause unexpected side effects?

  ***Result: An incorrect mouse picked does not cause any side effect.***

- Does the window close properly?

  ***Result: A window is closed down gracefully, either by clicking the exit/close/cancel button or the Close button (X) at the upper right corner.***

**For pull-down menus and mouse operations:**

- Is the appropriate menu bar displayed in the appropriate context?

  ***Result: All appropriate menu bars are displayed in the relevant windows.***

- Does the application menu bar display system-related features?

  ***Result: Application menu bars displayed only functions related to the window.***

- Do pull-down operations work properly?

  ***Result: All pull-down menu operations work as intended.***

- Are all menu functions and pull-down subfunctions properly listed?

  ***Result: All menu functions and pull-down subfunctions are properly listed when they are selected.***

- Are all menu functions properly addressable by the mouse?

  ***Result: All menu functions are addressable by the mouse.***

- Does tool bars work properly?

  ***Result: All tool bars worked very well.***

- Are text typeface, size and format correct?

  ***Result: Typeface, size and format are suitable to the context of its usage.***

- Is it possible to invoke menu functions using its alternative text-based command?

  ***Result: Menu functions are invokable by combination of Alt-(appropriate letter) command.***

- Are menu functions highlighted (or grayed-out) based on the context of current operations within a window?

  ***Result: Menu functions are highlighted or grayed-out depending on the context of its current operations.***

- Does each menu function perform as advertised?

  ***Result: Each of the menu functions performed as advertised.***

- Are the names of menu functions self-explanatory?

  ***Result: Unique and self-explanatory names are given to each of the menu functions to enhance understandability.***

- Does the cursor and processing indicator (e.g. an hourglass) properly change as different operations are invoked?

  ***Result: The cursor and processing indicator changed correspondingly.***

**Data entry:**

- Is invalid data properly recognized?

  ***Result: Invalid data entry is not permitted. Each input field is verified and validated during program development.***

- Are data input messages precise?

  ***Result: All data input messages are clear and accurate. Simple but precise words are used.***

- Do graphical modes of data entry (e.g. slide bars) work properly?

  ***Result: All graphical entry modes can be easily recognized, they functioned as intended.***

### 4.11.4 Help Facilities Testing

Errors in help facilities can be as devastating to the acceptance of the program as errors in data or source code. Nothing is more frustrating than following the facilities exactly and getting results or behaviors that do not coincide with those described by the facilities. For this reason, help facilities testing is carried out. It is approached in two phases. The first phase, help facilities are examined for editorial clarity and typing errors. The second phase, live test, the help facilities are used in conjunction with the use of the tool (two computers are installed with the same program and run in parallel). The following questions are used as guidelines [84], and program usage are tracked through the help documentation:

- Is the design of the document (layout, typefaces, indentation, graphics) conducive to understanding and quick assimilation of information?

  ***Result: The design of the document (Help and Tour Guide) are conducive and easy to understand.***

- Are the table of contents and index accurate and complete?

  ***Result: The table of content is clear and comprehensive.***

- Is it easy to locate guidance within the document?

  ***Result: It is easy to locate the required guidance. Separate topics are grouped according to their functions, and within each of the topics in ascending alphabetical order. Index search and hyperlinks are also available.***

- Are the terminology, menu descriptions and program responses consistent with the actual program?

  ***Result: All terminology and descriptions are standard software engineering terms. They are consistent with the actual program.***

- Is the description of each interaction sequence accurate?

  ***Result: Each interaction sequence accurately described the actual program order.***

- Does the documentation accurately describe how to accomplish each task?

  ***Result: Each task in the actual program is precisely described by the document. Tour Guide provides a step-by-step guidance on performing required tasks.***

- Can the troubleshooting be accomplished easily with the documentation?

  ***Result: Troubleshooting can be obtained from Help and Tour Guide.***

- Are examples accurate?

  ***Result: Help and Tour Guide use simple and understandable examples to guide users. Examples are relevant to the context.***

- Are the hypertext links accurate and complete?

  ***Result: All hypertext links worked well.***