FINAL YEAR PROJECT PAPER

DIPLOMA IN MECHANICAL ENGINEERING

FACULTY OF MECHANICAL ENGINEERING

MARA INSTITUTE OF TECHNOLOGY

SHAH ALAM SELANGOR D.E

ENGINEERING INTERACTIVE DESIGN USING INTERNET APPLICATION (PHASE 1)

PREPARED BY:

ROZAIDI SHAARIF
TENGKU AHMAD RASHIDI TENGKU NORDIN
MUHAMMAD ASRUL ZULKIFLI
(OKTOBER 1997)

ACKNOWLEDGEMENT

The authors would like to express their deepest gratitude and appreciation to En. Shaharudin Ahmad, our project supervisor, for his valuable contributions and suggestions towards the success of this project paper.

The authors are also sincerely grateful to

1. En. Abd. Malik
2. En. Shahrif

For their selflessness and priceless guidance in contributing for this project paper. However in addition to the person above, the author would also like to convey their upmost gratitude to the many individuals who helped towards the success of this project.

The authors hope that the project paper would assist and guide future Internet programming with much ease.

OBJECTIVE

This project, Engineering Interactive Design Using Internet (phase 1) is basically a introduction to the engineering students to the world of programming and publishing the program through the Internet. We are using Java WorkShop as a software to build an interactive program.

The program that we had produced is a basic interactive design where users can draw ovals. It is a kind like AutoCAD but it's much more simpler. Our program can be access through the Internet only because we made this program to use by the Internet surfers around the world.

Not only that, we are also introduces the way how to make a Web page and publish it through the Internet. So, to the Internet surfers that only surfs through the Internet, try make your own page.

CONTENTS

A BRIEF HISTORY OF SOFTWARE DEVELOPEMENT

We still deal with programming a line at a time because of the days when programs were written as a pile of cards and the cards were run through a card reader. In those days, programs executed as cards were read, and program editing consisted of adding and removing cards. More sophisticated mainframes stored the cards temporarily on disk before they were run. Since computers were expensive and operating systems were primitive, they had to be shared among many people, one person at a time. Programs had to be designed to run and complete so next user could have the machine.

Online systems came next. Even though the first online systems were designed for mainframes, time-sharing was normally associated with minicomputers running operating systems like UNIX. Interactive programming was born. Programs were still written in text in line-oriented text editors like ed. As printing terminals gave way to video displays, real editors like Emacs and vi appeared.

The UNIX development environment consisted of a shell a bag of utility programs like diff, grep, lint, and adb, and a compiler like cc. Specialized tools like yacc made it easy to write sophisticated parsers. The 64-kilobytes memory limit of early minicomputers encouraged a small-is-beautiful approach. The programmer had to figure out clever ways to combine the UNIX utilities to get the job done. Fortunately, programming was so much simpler in those days, since a single program couldn't contain more than a few thousand lines of code and user interfaces were rarely more sophisticated than a simple conversational typewriter like scheme.

The personal computer revolution spawned its own editors; for example, WordStar (really a word processor) provided many programmers with a useful tool. Otherwise, the early personal computers were too much like early minicomputers to allow for much innovation in development tools. The pervasiveness of low-resolution graphics capabilities in these early PCs did lead to creation of more graphics program with very different demands of programming tools. Little could be done, however, to expand the kinds of tools being used – until 1981, when IBM PC, with more than 64 kilobytes of RAM, marked the entry of affordable machines in the market.

The revolutionary Borland development environment, TurboPascal, was first released in 1983. This was the first combined text editor and fast compiler in one product. Through a succession of releases, Borland extended and perfected the concept of the integrated development environment. By the release of TurboC 2.0 in 1988, the environment included projects to build multifile programs, an integrated editor, debugger, compiler, and online help with a finely tuned, text based, windowing user interface. Everything about the product was engineered to make the mechanics of building and testing a program effortless.

Integrated environments came late to the UNIX world because as UNIX has evolved, programmers have assembled and integrated their own environment. They use high-powered workstations with numerous windows running different tasks. Although there's more typing involved, an ingenious UNIX hacker can customize shell scripts and makefiles to do almost anything. The flexibility is unbounded, even it does take a long time to put it all together.

Integrated environments, is contrast, use an all in one approach, and if you don't like one of the tools, you really can't move easily to another tool without switching the whole environment. Years of tool development, however, have made the PC environments very effective and easy to use.

The Java WorkShop design draws from the combined wisdom or the UNIX and the PC worlds. Here, Java has proven itself invaluable. Because Java WorkShop is written in Java, tools written as applets can share project information, allowing for sophisticated integration. Adding new tools is as easy as writing a new applet. Collaborative groups of applets can leverage the capabilities of Java to bring all kinds of new functionality for you. What you see today is only the beginning.