VIRTUAL ELECTRICAL TECHNOLOGY DEVELOPMENT CENTER

PROJECT REPORT

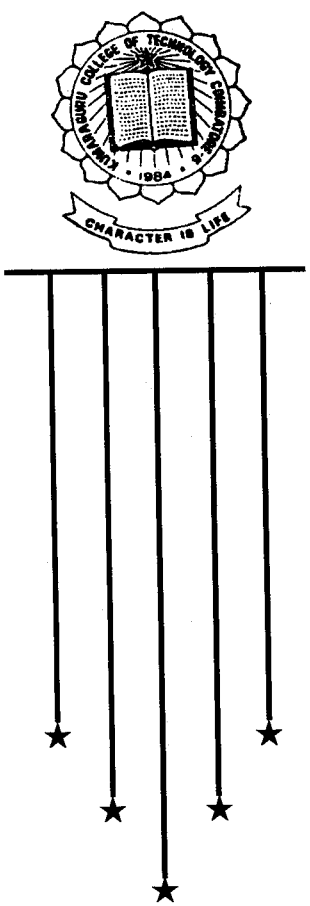
SUBMITTED BY

BINEESH. K

BISHU THOMAS

REJO ISON

SANGEETH KUMAR. M



1999 -2000

GUIDED BY

Mr. Dr. K.A. PALANISWAMY,

M.Sc.(Engg.), Ph.D., MISTE., C.Eng(I)., FIE.,

Prof. & Head of the Department.

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING IN

ELECTRICAL AND ELECTRONICS ENGINEERING

OF THE BHARATHIAR UNIVERSITY, COIMBATORE.

Department of Electrical and Electronics Engineering

Kumaraguru College of Technology

Coimbatore - 641 006.

CERTIFICATE

Kumaraguru College of Technology

Department of Electrical and Electronics Engineering

Coimbatore - 641 006.

CERTIFICATE

This is to certify that the contents of the project report entitled

VIRTUAL ELECTRICAL TECHNOLOGY DEVELOPMENT CENTER

has been submitted by Mr .BIBIN THOMAS, K.BALAJI, REJO JOSON, SANJEEV KUMAR

In Partial Fulfilment of the Requirements for the award of the

Degree of Bachelor of Engineering

IN ELECTRICAL AND ELECTRONICS ENGINEERING

Branch of the Bharathiar University, Coimbatore.

During the academic year 1999 - 2000

.....
Guide

Dr. K. A. PALANISWAMY, B.E., M.Sc.(Engg.), Ph.D., M.I.E., C.Eng.(I), F.I.E.
Professor and Head
Department of Electrical and Electronics Engineering
Professor And Head
Kumaraguru College of Technology.
Coimbatore 641 006

Certified that the canditate with University Register Number .....

was examined by us in project Viva - Voce Examination held on .....

.....
Internal Examiner

.....
External Examiner

Logic Version
TECHNOLOGIES

41-B, Bashyakaralu Road (West)
R. S. Puram, Coimbatore - 641 002, INDIA
Tel : 0422 - 434320
E-mail : lversion@giasmd01.vsnl.net.in

Coimbatore
10th March 2000

To Whomsoever it may concern

This is to certify that the following students of Kumaraguru College of Technology has completed their project " **Virtual Electrical Technology Development Center** ".During the period of their project work we found them dedicated , punctual and hard working.

We wish them a very good career.

Mr.K.Bineesh

Mr.Bishu Thomas

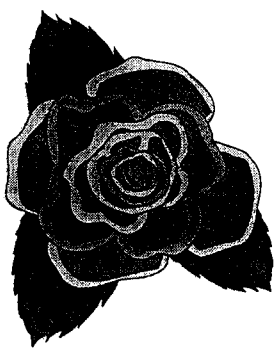
Mr.Rejo Ison

Mr.M.Sangeeth Kumar

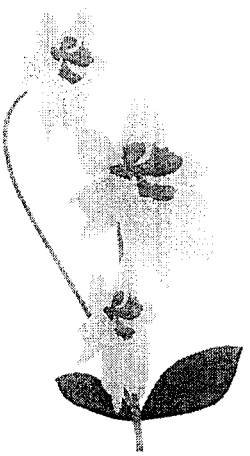
With regards
For Logic Version Technologies

Senthil V.K
Chief Executive Officer

P.K.Santharaam
Software Consultant



DEDICATED TO OUR PARENTS
WHO SACRIFICED THEIR TODAY FOR
OUR BETTER TOMORROW...



ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We express our sincere thanks and profound gratitude to our guide and Professor & Head, Department of Electrical & Electronics Engg. **Dr.K.A.PALANISWAMY**, BE, MSc (Engg) Ph.D, MISTE, C.Eng(I),F.I.E. for his valuable guidance & immense help at all stages of this project work.

We wholeheartedly express our deep gratitude to our Principal, **Dr.K.K.PADMANABHAN**, B.Sc,(Engg), M.Tech,Ph.D, for his constant encouragement.

We express our thanks to **Mr.P.K.SANTHARAAM**, B.E, Software Consultant, for permitting us to do the project in **LOGIC VERSION TECHNOLOGIES** , CBE.

We also wish to acknowledge the dynamic support of all our staffs, friends and parents who have helped us in many ways during the course of this project.

SYNOPSIS

SYNOPSIS

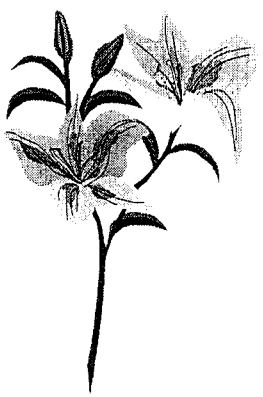
The prime objective of the project is to develop a web application for machine design and some other calculation. This web application runs under the web site which is aimed at the electrical community .The application is powered by the JAVA SERVLET ENGINE.

Since the application has been developed in Java Servlets there is no need for the net surfers to download the programs to perform the calculation, and the output design specification can be obtained in the HTML format instantly. This is the one of the prime features of the project. Apart from this, site also has various articles on Electrical and Electronics Engineering . It also has web pages on EEE jobs ,trends and many more.

The users can also publish articles on various EEE topics. The database is also maintained for members of the site. A general chat center is available where real time on-line discussions can be

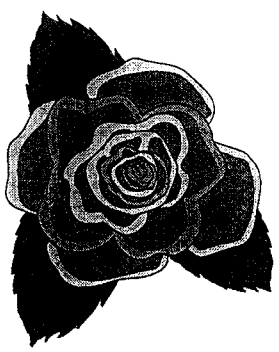
made .On-line seminars and techno group discussions can also be made. Private rooms can also be arranged on prior registration.

Thus this application as a project is aimed at the primary electrical community.



CONTENTS

CONTENTS



INTRODUCTION

CHAPTER I

INTRODUCTION

1.1 Introduction

The project, virtual electrical technology development center comprises of a web application which deals with the operation of electrical community this site has paper on EEE jobs. Recent treads and technical operations like machine design.

The company which is sponsoring and hoisting the site is logic version technologies, Coimbatore. They are pioneers in the field of web development and hoisting their hoisting and maintaining different sites under the domain logic version.com. The project is residing under domain electrix.com at electrix.i-p.com.

1.2 Software used to develop the web applications

The site has been developed with HTML, XML and Java script. The electrical applications are written and developed using Java servlets which is one of the powerful web output engines. Main advantage of using servlets is that the operations becomes more robust than that being developed with applets. Moreover considerably higher band width of data transmission is required for this system. Also the security of the system becomes quiet rigid.

The data required for design process is taken from a table residing in a database in the database server. The connection to the database from the main design calculations is made through Java Data Base Connectivity (JDBC) which follows the standard for database connectivity with Java.

1.3 Reason for using Java as a development tool.

Java is used as the development tool for this project in view of its platform independence, high level security, dynamic software distribution and object orientation.

The reason for making this operations on line is that the design calculations can be made available from anywhere in the world. This is not possible stand along application. Moreover the exposure to the technology is widened because of the increasing popularity and usage of cyber space.

1.4 The electrical calculations

Using servlets the calculations are done on-line machine application is designed in a way to receive data for calculation as input. The data is processed using the procedure. The output of design sheet is output usage servlets. The application are designing of DC machine. This report deals with the design of DC machine completely.

ANALYSIS DESIGN

CHAPTER 2

ANALYSIS AND DESIGN

The object-oriented paradigm is a new and different way of thinking about programming and many have trouble at first knowing how to approach a project. Now that you know that everything is supposed to be an object, you can create a "good" design by adopting following steps.

2.1 Staying on course

While you're going through the development process, the factors you are going to discover in this are

1. What are the objects?
2. What are their interfaces ?

The process can be undertaken in four phases, and a phase 0 which is just the initial commitment to using some kind of structure.

2.1.1 Phase 0: Let's make a plan

The first step is to decide what steps you're going to have in your process. Having a few milestones along the way helps to focus and galvanize our efforts around those milestones instead of being stuck with the single goal of "finish the project." In addition, it divides the project into more bite-sized pieces and make it seem less threatening. Assign the project modules systematically to the project members and see to that it is completed effectively.

2.1.2 Phase 1: What are we making?

The requirements analysis says, to make a list of the guidelines we will use to know when the job is done and the

customer is satisfied It's necessary to stay focused on the heart of what you're trying to accomplish in this phase: determine what the system is supposed to do. The most valuable tool for this is a collection of what are called "use-cases.". There have been a lot of attempts to come up with accurate scheduling techniques, but probably the best approach is to rely on your experience and instinct

2.1.3 Phase 2: How will we build it?

In this phase you must come up with a design that describes what the classes look like and how they will interact. A useful diagramming tool that has evolved over time is the Unified Modeling Language (UML). You can get the specification for UML at www.rational.com. UML can also be helpful as a descriptive tool during phase 1. and some of the diagrams you create there will probably show up unmodified in phase 2. The true beauty of the process was that the team learned how to do object-oriented design not by reviewing abstract examples, but by

suggests that you've actually built a pristine program and that all you need to do is change parts. The object-oriented programming languages are particularly adept at supporting this kind of continuing modification the boundaries created by the objects are what tend to keep the structure from breaking down. With iteration, you create something that at least approximates what you think you're building, and then you kick the tires, compare it to your requirements and see where it falls short. Iteration is closely tied to incremental.

JAVA AND INTERNET

CHAPTER 3

JAVA AND INTERNET

Java is a computer programming language. It is a general purpose object oriented programming language developed by Sun Microsystems, USA in 1991. It will solve traditional stand-alone programming problems and also programming problems on the World Wide Web(www).The main attributes of this language are

1. Compiled and interpreted
2. Platform independent and portable
3. Object-oriented
4. Robust and secure
5. Familiar, simple and small
6. Multithreaded and interactive
7. High performance
8. Dynamic and extensible

3.1 What is the Web?

Web is an interconnection of networks. Information on any subject can be got from the net. The terms surfing, presence and homepages are related to the web. The web works on the same manner as a Local Area Network (LAN),but in a very wide aspect ,but to do this you must understand client/server systems, another issue of computing .

3.2 Client/Server computing

The primary idea of a client/server system is that you have a central repository of information some kind of data, typically in a database that you want to distribute on demand to some set of people or machines. A key to the client/server concept is that the repository of information is centrally located so that it can be changed and so that those changes will propagate out to the information consumers. Taken together, the information repository, the software that distributes the information and the

machine(s) where the information and software reside is called the server. The software that resides on the remote machine, and that communicates with the server, fetches the information, processes it, and displays it on the remote machine is called the client.

Generally a database management system is involved so the designer "balances" the layout of data into tables for optimal use. In addition, systems often allow a client to insert new information into a server. This means you must ensure that one client's new data doesn't walk over another client's new data, or that data isn't lost in the process of adding it to the database. (This is called transaction processing.)

As client software changes, it must be built, debugged and installed on the client machines, which turns out to be more complicated and expensive. It's especially problematic to support multiple types of computers and operating systems. You might have hundreds of clients making requests of your server at any one time, and so any small delay is crucial. To minimize latency,

programmers work hard to offload processing tasks, often to the client machine but sometimes to other machines at the server site using so-called middleware. (Middleware is also used to improve maintainability.)

So the simple idea of distributing information to people has so many layers of complexity in implementing it that the whole problem can seem hopelessly enigmatic. Client/server computing accounts for roughly half of all programming activities.

It's responsible for everything from taking orders and credit card transactions to the distribution of any kind of data stock market, scientific, government – you name it. What we've come up with in the past is individual solutions to individual problems, inventing a new solution each time. These were hard to create and hard to use and the user had to learn a new interface for each one. The entire client/server problem is solved in a big way. It is the web server.

the Internet because any time you needed to do something that required programming you had to send information back to the server to be processed.. Since the browser was just a viewer it couldn't perform even the simplest computing tasks. To solve this problem, different approaches have been taken. To begin with, graphics standards have been enhanced to allow better animation and video within browsers. The remainder of the problem can be solved only by incorporating the ability to run programs on the client end, under the browser. This is called client-side programming .

3.4 Client-side programming

The Web's initial server-browser design provided for interactive content, but the interactivity was completely provided by the server. The server produced static pages for the client browser, which would simply interpret and display them. Basic HTML contains simple mechanisms for data gathering: text-entry boxes, check boxes, radio boxes, lists and drop-down lists, as well

as a button that can only be programmed to reset the data on the form or "submit" the data on the form back to the server. This submission passes through the Common Gateway Interface (CGI) provided on all Web servers. The text within the submission tells CGI what to do with it. The most common action is to run a program located on the server in a directory that's typically called "cgi-bin." These programs can be written in most languages.

Perl is a common choice because it is designed for text manipulation and is interpreted, so it can be installed on any server regardless of processor or operating system. Many powerful Web sites today are built strictly on CGI, and you can in fact do nearly anything with it. The problem is response time. The response of a CGI program depends on how much data must be sent as well as the load on both the server and the Internet. The initial designers of the Web did not foresee how rapidly this bandwidth would be exhausted for the kinds of applications people developed. It was very slow and not elegant..

The solution is client-side programming. Most machines that run Web browsers are powerful engines capable of doing vast work. Client-side programming means that the Web browser is harnessed to do whatever work it can, and the result for the user is a much speedier and more interactive experience at your Web site. A Web browser is like a limited operating system. In the end, it's still programming and this accounts for the dizzying array of problems and solutions produced by client-side programming.

The following are the approaches and issues in the client side programming.

(1) Plug-ins

This is a way for a programmer to add new functionality to the browser by downloading a piece of code that plugs itself into the appropriate spot in the browser. It tells the browser "from now on you can perform this new activity". The merit of plug-in for client-side programming is that it allows an expert programmer to

develop a new language and add that language to a browser without the permission of the browser manufacturer . Thus plug-ins provide the back door that allows the creation of new client-side programming languages.

(2) Scripting languages

With a scripting language you embed the source code for your client-side program directly into the HTML page and the plug-in that interprets that language is automatically activated while the HTML page is being displayed. Scripting languages tend to be reasonably simple to understand, and because they are simply text that is part of an HTML page they load very quickly as part of the single server hit required to procure that page. Scripting languages solve specific types of problem such creating a richer and more interactive graphical user interfaces. Solution such as Java or ActiveX programming seem to be considerable before going for scripting languages ,which are easier and more faster to develop. The most commonly-discussed scripting languages are JavaScript), VB Script and Tcl/Tk. JavaScript is probably the most

commonly supported. It comes built into both Netscape Navigator and the Microsoft Internet Explorer (IE).

(3) Java

The problems unsolved by the srerver-side programming is solved by Java. Java is being continuously extended to provide language features and libraries that elegantly handle problems that are difficult in traditional programming languages, such as multithreading, database access, network programming and distributed computing. Java allows client-side programming via the applet . An applet is a mini-program that will run only under a Web browser. When the applet is activated it executes a program. This is part of its beauty – it provides you with a way to automatically istribute the client software from the server at the time the user needs the client software, and no sooner. They get the latest version of the client software without fail and without difficult re-installation. The programmer needs to create only a single program, and that program automatically works with all computers that have browsers with built-in Java interpreters. Since

Java is a full-fledged programming language, you can do as much work as possible on the client before and after making requests of the server. Not only do you get the immediate win of speed and responsiveness, but the general network traffic and load upon servers can be reduced, preventing the entire Internet from slowing down.

One advantage a Java applet has over a scripted program is that it's in compiled form, so the source code isn't available to the client.. Two other factors can be important. As you will see later in the book, a compiled Java applet can comprise many modules and take multiple server "hits" (accesses) to download. A scripted program will just be integrated into the Web page as part of its text This could be important to the responsiveness of your Web site. Regardless of what you've heard.

(4) ActiveX

To some degree, the competitor to Java is Microsoft's ActiveX. ActiveX is originally a Windows-only solution, although it is now being developed via an independent consortium to

become cross-platform. If your program connects to its environment easily, it can be dropped into a Web page and run under a browser that supports ActiveX. IE directly supports ActiveX and Netscape does so using a plug-in. ActiveX does not constrain you to a particular language. Programs in C++, Visual Basic, or Borland's Delphi, you can create ActiveX components with almost no changes . ActiveX also provides a path for the use of legacy code in your Web pages.

3.5 Security

Automatically downloading and running programs across the Internet can allow virus's to get into your system. ActiveX especially brings up the thorny issue of security in client-side programming. Java was also designed to run its applets within a "sandbox" of safety, which prevents it from writing to disk or accessing memory outside the sandbox.

ActiveX is at the opposite end of the spectrum. Programming with ActiveX is like programming Windows – you can do anything you want.

So if you click on a page that downloads an ActiveX component, that component might cause damage to the files on your disk. Of course, programs that you load onto your computer that are not restricted to running inside a Web browser can do the same thing. The solution seems to be "digital signatures," whereby code is verified to show who the author is . This is based on the idea that a virus works because its creator can be anonymous, so if you remove the anonymity individuals will be forced to be responsible for their actions. This seems like a good plan because it allows programs to be much more functional, and I suspect it will eliminate malicious mischief.

The Java approach is to prevent these problems from occurring, via the sandbox. The Java interpreter that lives on your local Web browser examines the applet for any untoward instructions as the applet is being loaded. In particular, the applet cannot write files to disk or erase files. Applets are generally considered to be safe, and since this is essential for reliable client-server systems, any bugs that allow viruses are rapidly repaired.

Java 2 provides a framework for digital signatures so that you will eventually be able to allow an applet to step outside the sandbox if necessary.

3.6 Internet vs Intranet

The Web is the most general solution to the client/server problem, so it makes sense that you can use the same technology to solve a subset of the problem, in particular the classic client/server problem within a company. With traditional lient/server approaches you have the problem of multiple different types of client computers, as well as the difficulty of installing new client software, both of which are handily solved with Web browsers and client-side programming. When Web technology is used for an information network that is restricted to a particular company, it is referred to as an Intranet. Intranets provide much greater security than the Internet, since you can physically control access to the servers within your company The security problem brings us to one of the divisions that seems to be automatically forming in the world of client-side programming. If your program is running on

the Internet, you don't know what platform it will be working under and you want to be extra careful that you don't disseminate buggy code. You need something cross-platform and secure, like a scripting language or Java. If you're running on an Intranet, you might have a different set of constraints. It's not uncommon that your machines could all be Intel/Windows platforms.

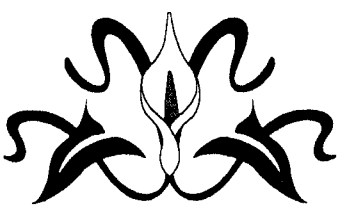
On an Intranet, you're responsible for the quality of your own code and can repair bugs when they're discovered. In addition, you might already have a body of legacy code that you've been using in a more traditional client/server approach, whereby you must physically install client programs every time you do an upgrade. The time wasted in installing upgrades is the most compelling reason to move to browsers because upgrades are invisible and automatic. If you are involved in such an Intranet, the most sensible approach to take is ActiveX rather than trying to recode your programs in a new language.

3.7 Server-side programming

What happens when you make a request of a server is, your browser interprets the file in some appropriate fashion: as an HTML page, a graphic image, a Java applet, a script program, etc. A more complicated request to a server generally involves a database transaction. A common scenario involves a request for a complex database search, which the server then formats into an HTML page and sends to you as the result or you might want to register your data into a database, which will involve changes to that database.These database requests must be processed via some code on the server side, which is generally referred to as server-side programming . Traditionally, server-side programming has been performed using Perl and CGI scripts, but more sophisticated systems have been appearing. These include Java-based Web servers that allow you to perform all your server-side programming in Java by writing what are called servlets.

problems like networking and cross-platform UI, and yet it has a language design intended to allow the creation of very large and flexible bodies of code. Add to this the fact that Java has the most robust type checking and error-handling systems It has a special feature of significant leap forward in programming

Why we use Java instead of C++ for your project? There are two issues to consider. First, if you want to use a lot of existing libraries , or if you have an existing C or C++ code base, Java might slow your development down rather than speeding it up. If you're developing all your code primarily from scratch, then the simplicity of Java over C++ will shorten your development time. The biggest issue is speed. Interpreted Java has been slow, even 20 to 50 times slower than C in the original Java interpreters. This has improved quite a bit over time, but it will still remain an important numberThe key to making Java feasible for most non-Web development projects is the appearance of speed improvements like so-called "just-in time" (JIT) compilers and possibly even native code compilers (two of which exist at this writing). Of course, native-code compilers will eliminate the touted cross-



WEB SITE

CHAPTER 4

WEB SITE

The web application is presented as web site. This web site unlike ordinary web site is dynamic which means it can interest with the user. The interaction is mainly in the form of an electrical design procedure, which receives inputs and gives output. The other features include a discussion center, feedback and many more each feature of the web site is designed below briefly.

4.1 Home page

When you log on the site using the URL – (www.electrix.i-p.com) you directly go to the home page. This is the heart of the web site. This web page has many links which are different facilities of feature of applications. There links take you to different pages which contain different purposes of its on. Each link has on option called Back which takes back to the home page.

The home page contains many interesting facts, hot news, 'gif' images etc. which make the home page attractive.

The various like from this web site are described below.

4.1.1 Profile

This link from home page takes you to a page where you get details about the developers of the this project and other related details. Persons who where technically involved in the project are also presented in this page.

4.1.2 About EEE center

This page gives the information about what this project is all about. It introduces you to the various technologies utilized and facilities available on the site. It gives an introduction about this project.

4.1.3 References

This link page is a library which contains details of electrical books, their authors and other details. The library contains about 10,000 books titles and authors.

4.1.4 Information

This page gives hot news in electrical field. They also contains articles submitted for seminars and paper presentation. This gives one, Idea about future trends and technical know how about new technologies etc. This page is updated every week to get the site more alternative and a new look.

4.1.5 EEE Trends

This link when enabled leads you to a new web site similar to one site. Initially it is programmed to lead one to a new site called electricmela.com. This link will be changed depending upon public demand and many new sites coming up.

4.1.6 Discussion center

Discussion center is a basic chat room, where you can log on and chat with others logged on. This application can be used in following ways :-

1. As a platform for conducting a group discussion center.
2. As a medium for conducting a seminar.
3. It is also a entertain center for technical people.

4.1.7 Site Watch

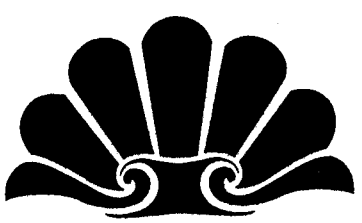
This page gives reference to many web sites of similar nature.

It only have a library containing URL of different sites.

4.1.8 EEE jobs

This page contains information and details about various career options and placement supports available.

The facility used in the site are more useful for technical community and well, can be specified as a **'Virtual Electrical Technology Development Center'.**



WEB APPLICATION

CHAPTER 5

WEB APPLICATION

Consider creating an application to run on the Web, which will show Java in all its glory. Part of this application will be a Java program running on the Web server, and the other part will be an applet that's downloaded to the browser. The applet collects information from the user and sends it back to the application running on the Web server. The task of the program will be simple: the applet will ask for the email address of the user, and after verifying that this address is reasonably legitimate (it doesn't contain spaces, and it does contain an '@' symbol) the applet will send the email address to the Web server. The application running on the server will capture the data and check a data file in which all of the email addresses are kept. If that address is already in the file, it will send back a message to that effect, which is displayed by the applet. If the address isn't in the file, it is placed in the list and the applet is informed that the address was added successfully.

5.1 The Server Application

Consider the server application, which will be called **NameCollector**. If more than one user at a time tries to submit their email addresses, then it must use the multithreading approach to handle more than one client at a time. But all of these threads will try to write to a single file where all the email addresses will be kept. This would require a locking mechanism to make sure that more than one thread doesn't access the file at once. The **Name Collector** uses TCP/IP sockets.

If we use datagrams instead, multithreading is unnecessary. A single datagram socket will listen for incoming datagrams, and when one appears the program will process the message and send the reply as a datagram back to whomever sent the request. If the datagram gets lost, then the user will notice that no reply comes and can then re-submit the request.

When the server application receives a datagram and unpacks it, it must extract the email address and check the file to see if that address is there already.

5.1.1 The NameSender applet

As mentioned earlier, the applet must be written with Java 2.0 so that it will run on the largest number of browsers, so it's best if the number of classes produced is minimized. Thus, instead of using the **Dgram** class developed earlier, all of the datagram manipulations will be placed in line. In addition, the applet needs a thread to listen for the reply from the server, and instead of making this a separate thread it's integrated into the applet by implementing the **Runnable** interface.

5.1.2 Problems with this approach

This certainly is an elegant approach. There's no CGI programming and so there are no delays while the server starts up a CGI program. The datagram approach seems to produce a nice quick response. In addition, when Java 2 is available everywhere, the server portion can be written entirely in Java.

There are problems, however. One problem is rather subtle: since the Java application is running constantly on the server and it spends most of its time blocked in the **Datagram.receive** method, there *might* be some CPU hogging going on.

The second problem is a show stopper. It concerns firewalls. A firewall is a machine that sits between your network and the Internet. It monitors all traffic coming in from the Internet and going out to the Internet, and makes sure that traffic conforms to what it expects. Firewalls are conservative little beasts. They demand strict conformance to all the rules, and if you're not conforming they assume that you're doing something sinful and shut you out . As long as your customers have raw connections to the Internet there's no problem, but you might have some important customers dwelling behind firewalls, and they won't be able to use your program.

Web servers will be equipped with *servlet servers* . These will take a request from the client and instead of starting up a CGI

program they will start up a Java program called a *servlet*. This is a little application that's designed to run only on the server. A servlet server will automatically start up the servlet to handle the client request, which means you can write all your programs in Java.

5.2 Identifying a Machine

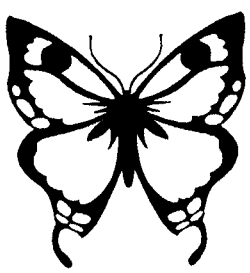
Of course, in order to tell one machine from another and to make sure that you are connected with the machine you want, there must be some way of uniquely identifying machines on a network. Early networks were satisfied to provide unique names for machines within the local network. However, Java works within the Internet, which requires a way to uniquely identify a machine from all the others in the world . This is accomplished with the IP (Internet Protocol) address that can exist in two forms:

1. The familiar DNS (Domain Name Service) form. And is often incorporated into a World-Wide-Web address.

2. Alternatively, you can use the "dotted quad" form, which is four numbers separated by dots, such as 123.255.28.120.

and from then on you can treat the connection as if you were reading from and writing to a file. This is one of the nice features of Java networking.

An IP address isn't enough to identify a unique server, since many servers can exist on one machine. Each IP machine also contains ports, and when you're setting up a client or a server you must choose a port where both client and server agree to connect. The port is not a physical location in a machine, but a software abstraction. The client program knows how to connect to the machine via its IP address. That's where the port numbers come in as second level of addressing. The idea is that if you ask for a particular port, you're requesting the service that's associated with the port number. The time of day is a simple example of a service. Typically, each service is associated with a unique port number on a given server machine. It's up to the client to know ahead of time which port number the desired service is running on.



DESIGN OF DC MACHINE

CHAPTER 6

DESIGN OF D.C MACHINES

6.1 Introduction:

D.C Machine can work has motor, generator and brakes. In generator mode the machine is drawn by a prime mover with the mechanical power converted into electoral power. In motor mode the machine drives a mechanical load with the electrical power supplied converted into mechanical power. In brake mode the machine decelerates an account of the power supplied or dissipated by it and therefore produces a mechanical braking action.

6.2 Classification:

1. DC Generator
2. DC Brakes

6.4 Constructional Details:

The DC Commutator machines used for industrial application have three major Parts.

1. Field system
2. Armature
3. Commutator

6.4.1 Field System:

The Field system is located on the stationary part of the machine called Stator. The field system is designated for producing magnetic flux and therefore provides necessary excitation for operation of machine.

The Stator consists of

1. Main Poles: Three Poles design to produce the main magnetic flux. The are made of laminated Steel.

2. Inter Poles: These are placed between the main poles to improve commutation to ensure spark less operation. They are made from laminated Steel or Low Cost Carbon Steel.
3. Frame: This Provide support for the machine. This is made using cast Steel.

6.4.2 Armature:

The Armature of DC machine is build up of Thin Lamination Low Loss Silicon Steel. In Small machines the armature laminations are filled directly on the shaft and are clamped tightly between end flanges which also act as supports for the armature winding. Medium machines have their laminations build upon a spider. The Spider is casted are fabricated. In Large machines dove tailed laminations are fit into similar shape spiders.

Armature winding:

In DC machines two layer winding with diamond shaped coils is used. In small machines steel wire and in large machines wedges are used to keep the winding in Position .

6.4.3 Commutator:

The function of the commutator is to rectify the alternating current induced in the alternators. The essentials of the construction are a number of copper bars or segments insulated from each other and connected to armature conductors. The Commutator is made of silvered copper.

6.5 DESIGN PROCEDURE

The following inputs are taken to design the machine :

Type of machine to be designed -(motor or generator)

Voltage rating v(volts)

Power rating p(kw)

No of poles po

Speed n(rpm)

Flux density art air gap bg (Wb/m2)

Average flux density bav(Wb/m2)

Ampere conductors ac(A/m)

6.5.1 MAIN DIMENSION

Efficiency of the machine is assumed to be 98%

For motor power generated pa=(1+2*0.92)/(3*0.92) (W)

For generator pa=p/0.92 (W)

Synchronous speed ns=n/60 (rps)

Output co-efficient co=(3.14)*(3.14)*bav*ac*0.001

The ratio of pole pitch to pole arc lt=0.67

Diameter and length of the core is solved from the simultaneous equation given below

D*D*L=pa/(co*ns)

L=lt*(3.14)*D/po (m)

Pole pitch t=(3.14)*D/po (m)

Pole arc tarc=lt*L (m)

Peripheral speed va=(3.14)*D*ns (m/s)

Frequency f=po*ns/2 (hz)

Nett iron length li=0.9(l-0.01) (m)

Thickness of lamination =0.35 (mm)

6.5.2 ARMATURE DESIGN

Termial voltage =240(v)

Line current il=p*1000/(0.92*v) (A)

Percentage field current(if) and inetrnal voltage drop(vi) are choosen corresponding to value of the prpoduct pa*n

Field current iff=il*if/100 (A)

Internal vol;tage drop vii=vi*v/100 (V)

Armature current ia=il-iff (A)

Generated voltage e=vi-vii (V)

If ia <=400----wave winding is used

Number of parrel path a=2

Else ----------lap winding is used

Number of parallel path a=p

Current per parellel path ipp=ia/(A)

Flux per pole fp=0.5*lt6(wb)

Number of armature conductors z=e*a/(ns*po*fp)

Number of slots

The number armature slot lies between 3.14*d/35 and 3.14*d/25

Slots per pole should not be divisible by p0/2 select the largest

Put u as 2,4,6.....

Such that c=1/2u*s*s should be greater than e*p0/15

Number of armature conductors actually used z=2c

Number of armature conductors per slot zs=z/s

Slot pitch ys=3.14*D/s (m)

Modified flux per pole fp1=fp*z/z1 (Wb)

Actual bav=fp1/(t*l) (Wb/mm2)

Actual ac =ipp*z1/(3.14*D) (A/m)

6.5.3 DESIGN OF SLOT

Current density used in the armature winding da=6 (A/mm2)

Area of armature conductor aac=ipp/da (m2)

Depending on the area of the conductor the dimension (ws X ds)

of the conductor is chosen

slot width (ws) is found considering conductor insulation ,slot insulation and slack together with conductor size

lsot depth (ds) is found cosidering seperator ,lipand wedge together with the above parameters

6.5.3 LENGTH OF AIR GAP

Armature mmf per pole ata=ipp*z/(a*p)

The mmf required for air gap is assumed to be 0.6 times the armature mmf

Mmf at air gapatg=0.6*ata

Maximum flux density in the air gapbg=bav/t

Gap contraction factorkg=1.15

Length of air gap lg =atg/(80000*bg*kg)

6.5.4 ARMATURE CORE

Flux in armature core fac=fp1/2

Flux density in the core is assumed to be 1.2

Area of armature coreac=fac/1.2

Depth of armature core dc=ac/li

Internal diameterdi=D-2(dc+dc)

FIELD SYSTEM WINDING

6.5.5 POLE SECTION

Leakage co-efficient cf=1.15

Flux in the pole body fpp=fp1*1.15

Flux density in the pole bodyis assumed to be bp=1.5

Area of the pole body ap=fpp/1.5

The axial length of the pole body is same as the armature lp=L

Nett iron length of the pole bnody lpi=0.9*lp

Width of pole body wp=ap/lpi

6.5.6 TENATIVE DESIGN OF FIELD WINDING

The field mmf at fullload is assumed to be 90% of aramture mmf

Field mmf atfl=0.9*ata

Assumptions made --------space factor sf=0.68

Loss decipation qf=700

Depth of winding df=45mm

Mmf per meter of winding atwh=10000*sqr(qf*sf*df)

Hieght of field winding hf=atfl/atwh

Hieght of pole shoe hps=20mm

Lamination thickness hi=20 mm

Hieght of pole hp=ht+hps+hi

6.5.7 YOKE

Flux in the yoke fy=fp1/2

Flux densityby=1.5

Area of yoke ay=fy/by

Depth of yoke dy=ay/lpi

Outer diameter of yoke dy=d+2(lg+hpi+dy)

6.5.8 MAGNITIC CIRCUIT

Ratio of slot opening to gap length is foundout corresponding to it Carters co-efficient is found out .gap contraction factor can be calculated from this. The sane procedure is employed for finding the gap contraction factor for ducts.

Total gap contracton factor ks=kgs*kgd

Mmf for air gap atg=80000*lg*kg*bg

6.5.9 MMF OF TEETH

Pole pitch 1/3 hieght h3=3.14*(D-4/3*ds)/(s)

Width w3=ys3-ws

Flux density bt3=p*fp1/(t*s*lt*w3)

Corresponding to bt3 mmf/m(at) is selected

Mmf for teeth att=at*lt

6.5.10 MMF OF CORE

Flux desity bc

Length of core lc=3.14*(D-2*ds-dc)/(a*p)

Mmf/m(atc)is chosen corresponding to bc

Mmffor core attc=atc*lc

The same procedure is followed to find the mff of pole and yoke

Total mmf required attt=att+attc+atp+aty

At full load mmf is atfl=1.15*attt

6.5.11 DESIGN OF FIELD WINDING

voltage across each field coilef=(v-0.2*v)/p

Length of mean turn lmt=2*(lp+hp+2*dp)

Resistivity of copper res=0.021

Area of conductor af=atfl*res*lmt/v

Space factor sf=0.7(d/d+d1)2

Where d1 is selected depending on d

Number of turns tf=sf*df1*hf/af

Resistance of each coil rf=atf*res*lmt/af

Field current if=ef/rf

Field mmf provided atflp=if*tf

6.6 PROGRAM TO DESIGN A DC MACHINE USING JAVA

```
// DC Machine Design

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.sql.*;

public class ser992 extends HttpServlet
{
String v1,p1,n1,po1,bg1,bav1,ac1,hai,tw;
PrintWriter out;
double if1,vi,s,ipp,pa,c,u=2,a,atwh,r1,j,ajt,d1,bp=1.5,atflp,row,rf;
double v,p,bg,bav,ac,c0,g,b,d,l,t,tarc,va,li,il,x,e,bc,wp,df,f,tf,If;
double
f1,iff,vii,ia,fp,z,sp,cf,z1,zs,ys,fp1,bav2,c1,ac2,dac,aac,yc,ws,ds,ata,atg,lg,fac;
double
aa,dc,bi,di,fpp,ap,lp,lpi,atfl,qs,sf,atwn,hf,hp,fy,by,ay,dy,ddy,kcs,kgs,r2;
double
kcd,wd,kgd,kg,ys3,wt3,bt3,ks,br,atr,att,le,atc,atp,ly,aty,attt,atttfl,ef,lmt,af;
int n,po,ns;
long pop;

        public void init(ServletConfig con)throws ServletException
        {
        super.init(con);
        }

public void doGet(HttpServletRequest req,HttpServletResponse res)throws
ServletException,IOException
{
  res.setContentType("text/html");
  out=res.getWriter();
  out.println("<html><body> Results for your Data submitted:");

  Enumeration enu  = req.getParameterNames();
  out.println("<html><body><div        align=center><br><h3>        Design
Sheet</h3>");
```

```
    String str="hai",str2="";
    while(enu.hasMoreElements())
    {    str=(String)enu.nextElement();

  if(str.equals("D1")){  hai=req.getParameter(str);    }
  if(str.equals("T1")){  v1=req.getParameter(str);     }
  if(str.equals("T2")){  p1=req.getParameter(str);     }
  if(str.equals("T3")){  n1=req.getParameter(str);     }
  if(str.equals("T4")){  po1=req.getParameter(str);    }
  if(str.equals("T5")){  bg1=req.getParameter(str);    }
  if(str.equals("T6")){  bav1=req.getParameter(str);   }
  if(str.equals("T7")){  ac1=req.getParameter(str);    }
    }

  dob_cal( );

        if(hai.equalsIgnoreCase("motor"))
        {       pa=0.667*p;   }
        else
        {       pa=p/0.92;    }

  formula_cal( );

  compare( );

  cal_2( );

  rock( );

  success1( );

  success2( );

  success3( );

}
//-------------------------------------------------------------------------------
        public void dob_cal( )
        {
        Double f1=new Double(v1);
```

```
        Double f2=new Double(p1);
        Double f5=new Double(bg1);
        Double f6=new Double(bav1);
        Double f7=new Double(ac1);

        v=f1.doubleValue( );
        p=f2.doubleValue( );
        n=Integer.parseInt(n1);


        po=Integer.parseInt(po1);
      bg=f5.doubleValue( );
        bav=f6.doubleValue( );
        ac=f7.doubleValue( );
        }

//-------------------------------------------------------------------------------
        public void formula_cal( )
        {

     ns=n/60;
     c0= 0.00985* bav * ac;
        g=pa/(c0*ns);
        b=0.67*3.14/ po;
        d=Math.pow((g/b),(1.0/3.0));
        l=b*d;
        t=3.14*d/po;
        tarc=0.67*t;
        va=3.14*d*ns;
        f=(po*ns)/2;
        li=0.9*(l-0.01);
        il=1086.95*p/v;
        x=pa*ns*60;
        }
//-------------------------------------------------------------------------------

        public void compare( )
        {
        if(x<=10000 )                    { ifl= 2.5 ; vi=6.6 ;}
        else if((x>10000)&&(x<=20000))  { ifl=2.1  ; vi= 6;}
        else if((x>20000)&&(x<=30000))  { ifl= 1.9 ; vi=5.5 ;}
```

```
        else if((x>30000)&&(x<=40000))    { ifl= 1.8 ; vi=5.2 ;}
        else if((x>40000)&&(x<=50000))    { ifl= 1.6 ; vi=5 ;}
        else if((x>50000)&&(x<=60000))    { ifl= 1.5 ; vi= 4.9;}
        else if((x>60000)&&(x<=70000))    { ifl=  1.4; vi=4.8 ;}
        else if((x>80000)&&(x<=90000))    { ifl= 1.3 ; vi= 4.7;}
        else if((x>90000)&&(x<=100000))   { ifl= 1.2 ; vi= 4.6;}
        else if((x>100000)&&(x<=200000)){ ifl=1.1  ; vi=4.5 ;}
        else if((x>200000)&&(x<=300000)){ ifl=0.8  ; vi=4 ;}
        else if((x>300000)&&(x<=400000)){ ifl=0.6  ; vi=3.75 ;}
        else if((x>400000)&&(x<=500000)){ ifl= 0.5 ; vi=3.5 ;}
        else if((x>500000)&&(x<=600000)){ ifl=0.45  ; vi=3.25 ;}
        else if((x>600000)&&(x<=700000)){ ifl= 0.4  ; vi=3.1 ;}
        else if((x>700000)&&(x<=800000)){ ifl= 0.36 ; vi= 3;}
        else if((x>800000)&&(x<=900000)){ ifl= 0.33 ; vi=2.55 ;}
        else if((x>900000)&&(x<=1000000)){ ifl= 0.31; vi=2.8;}
        else { ifl= 0.33 ; vi=2.55 ;}
        }
//-------------------------------------------------------------------------------

public void cal_2( )
{
iff=2.5*ifl/100;
vii=6.6*vi/100;
ia=il-iff;
e=v-vii;
        if(ia<=400)
        {
        tw="wave winding";
        a=2;
        ipp=ia/2;
        }
        else
        {
        tw="lap winding";
        a=po;
        ipp=ia/a;
        }
fp=0.5*l*t;


z=(e*a)/(ns*po*fp);
```

```
sp=3.14*d/(25*po);

}

//-----------------------------------------------------------------------

public void rock( )
{
        for( ; ; )
        {
                if((sp%(po/2))!=0)
                {
                s=sp*po;
                break;
                }
        else
                {
                  sp=sp-1;
                }
        }

        c1=e*po/15;

        for( ; ; )
        {
                c=Math.pow((s),(2))*u/2;

                if(c<c1)
                {
                u=u+2;
                }
                else
                {
                break;
                }
        }
//-----------------------------------------------------------------------
 z1=2*c;
 zs=z1/s;
 ys=3.14*d/s;
```

```
fp1=fp*z/z1;
bav2=fp1/(t*l);
ac2=ipp*z1/(3.14*d);
dac=6000000;
aac=ia/dac;
yc=aac/10;
ws=(zs*aac/0.010)+zs*0.035+0.002+0.0005;
ds=0.02+4*0.035+zs/2+0.001+0.004+0.0006;
ata=ipp*z/(a*p);
atg=0.6*ata;
lg=atg/(80000*bg*1.15);
fac=fp1/2;
aa=fac/1.2;
dc=aa/li;
bc=fac/ac;
di=d-(2*(dc+ds));
fpp=fp1*1.15;
ap=fpp/1.5;
lp=l;
lpi=0.9*lp;
wp=ap/lpi;
atfl=0.9*ata;
qs=700;sf=0.68;df=0.045;
atwn=10000*Math.pow((qs*sf*df),(1/2));
hf=atfl/atwn;
hp=hf+0.003;
fy=fp1/2;
by=1.5;
ay=fy/1.5;
dy=ay/lpi;
ddy=d+(2*(lg+hp+dy));
r1=ws/lg;

kcs=tab1(r1);
kgs=ys/(ys-(kcs*ws));
r2=10/lg;

kcd=tab1(r2);
wd=10;

kgd=l/(l-(kcd*wd));
```

```
kg=kgs*kgd;
atg=80000*kg*lg*bg;
ys3=3.14*(d-(4*ds/3));
wt3=ys3-ws;
bt3=po*fpp/(li*wt3*t*s);
ks=l*ys3/(li*wt3);
br=bt3-(4*3.14*(ks-1));

atr=tab2(br);
att=atr*ds;

le=(3.14*(d-2*ds-dc))/(a*p);
atc=tab2(bc)*le;

atp=tab2(bp)*hp;

ly=0.6;
aty=tab2(by)*ly;

attt=atg+att+atp+aty+ata;
atttfl=1.115*attt;
ef=v-(0.2*v);

lmt=2*(hp+lp+2*bp);
af=(atttfl+lmt)/v;



if(d<=0.09){double d1=0.01;}
if((d>0.09)&&(d<=0.18)){double d1=0.02 ;}
if((d>0.18)&&(d<=0.3)){double d1=0.032 ;}
if((d>0.3)&&(d<=0.4)){double d1=0.045 ;}
if((d>0.4)&&(d<=0.5)){double d1=0.05;}
if((d>0.5)&&(d<=0.6)){double d1=0.054 ;}
if((d>0.6)&&(d<=0.95)){double d1=0.065 ;}
if((d>0.95)&&(d<=1.4)){double d1=0.075 ;}
if((d>1.4)&&(d<=2)){double d1=0.09 ;}
if((d>2)&&(d<=3)){double d1=0.110 ;}
if((d>3)){d1=0.125 ;}

sf=0.75*d/(Math.pow((d+d1),(2)));
```

```
 tf=sf*af*d/af;
 row=2;
 rf=tf*row*lmt/a;
 If=ef/rf;
 atflp=If*tf;
}
//-------------------------------------------------------------------------------
public double tab1(double r1)
{
if((r1<=1)){ j= 0.15;}
if((r1>1)&&(r1<=2)){ j=0.3 ;}
if((r1>2)&&(r1<=3)){ j=0.4 ;}
if((r1>3)&&(r1<=4)){ j=0.45 ;}
if((r1>4)&&(r1<=5)){j= 0.5;}
if((r1>5)&&(r1<=6)){ j= 0.55;}
if((r1>6)&&(r1<=7)){ j= 0.6;}
if((r1>7)&&(r1<=8)){ j=0.625 ;}
if((r1>8)&&(r1<=9)){ j=0.65 ;}
if((r1>9)&&(r1<=10)){ j=0.675 ;}
if((r1>10)&&(r1<=11)){ j= 0.7;}
if((r1>11)){j=0.715 ;}
return j ;
}
//-------------------------------------------------------------------------------
public double tab2(double b)
{
if((b<1.2)){ ajt=500 ;}
if((b>1.2)&&(b<=1.3)){ ajt=1000 ;}
if((b>1.3)&&(b<=1.4)){ ajt=2000 ;}
if((b>1.4)&&(b<=1.5)){ ajt=3000;}
if((b>1.5)&&(b<=1.6)){ ajt=4000 ;}
if((b>1.6)&&(b<=1.7)){ ajt=7500 ;}
if((b>1.7)&&(b<=1.8)){ ajt=10000 ;}
if((b>1.8)&&(b<=1.9)){ ajt=16000 ;}
if((b>1.9)&&(b<=2)){ ajt=17000 ;}
if((b>2)&&(b<=2.1)){ ajt=41000 ;}
if((b>2.1)&&(b<=2.2)){ ajt=74000 ;}
if((b>2.2)){ajt=110000 ;}
return ajt;
}
```

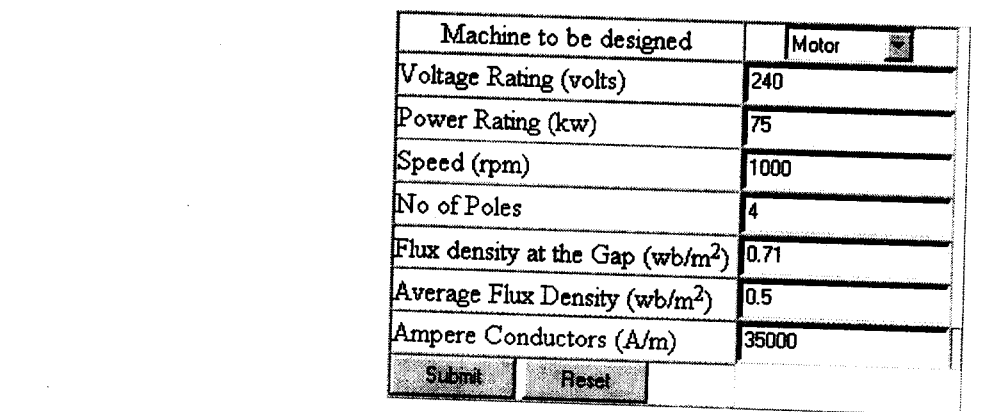
```
//-------------------------------------------------------------------------------
-------------------------
public void success1( )
{
out.println("<div align=center><center><table border=1 cellpadding=0
cellspacing=1>");
out.println("<tr><td><b>PARAMETERS</b></td><td>VALUES</td><td>
Units</td></tr>");
out.println("<tr><td>Voltage Rating </td><td>"+v+"</td><td>
Volts</td></tr>");
out.println("<tr><td>Power Rating </td><td>"+p+"</td><td>
KW</td></tr>");
out.println("<tr><td>Speed </td><td>"+n+"</td><td> rpm</td></tr>");
out.println("<tr><td>No of Poles </td><td>"+po+"</td><td> </td></tr>");
out.println("<tr><td>Flux Density</td><td>"+bg+"</td><td>
Wb/m<sup>2</sup></td></tr>");
out.println("<tr><td>Average Flux Density
</td><td>"+bav+"</td><td>Wb/m<sup>2</sup></td></tr>");
out.println("<tr><td>Ampere Conductors </td><td>"+ac+"</td><td>
A/m</td></tr>");
out.println("<tr><td>Power generated </td><td>"+pa+"</td><td>
(KW)</td></tr>");
out.println("<tr><td>Synchronous Speed </td><td>"+ns+"</td><td>
rps</td></tr>");
out.println("<tr><td>Output Co-efficient </td><td>"+c0+"</td><td>
</td></tr>");
out.println("<tr><td>Armature Diameter </td><td>"+d+"</td><td>
m</td></tr>");
out.println("<tr><td>Core Length </td><td>"+l+"</td><td> m</td></tr>");
out.println("<tr><td>Pole Pitch </td><td>"+t+"</td><td> m</td></tr>");
out.println("<tr><td>Pole Arc </td><td>"+tarc+"</td><td> m</td></tr>");
out.println("<tr><td>Peripheral Speed </td><td>"+va+"</td><td>
m/s</td></tr>");
out.println("<tr><td>Frequency </td><td>"+f+"</td><td>Hz </td></tr>");
out.println("<tr><td>Nett Iron Length </td><td>"+li+"</td><td>m
</td></tr>");
out.println("<tr><td>Line Current</td><td>"+il+"</td><td>Amps
</td></tr>");
out.println("<tr><td>Field Current</td><td>"+iff+"</td><td>Amps
</td></tr>");
out.println("<tr><td>Internal Voltage Drop
</td><td>"+vii+"</td><td>volts</td></tr>");
```

```
}
//-------------------------------------------------------------------------
public void success2( )
{
out.println("<tr><td>Armature Current </td><td>"+ia+"</td><td>
amps</td></tr>");
out.println("<tr><td>Generated EMF </td><td>"+e+"</td><td>
volts</td></tr>");
//out.println("<tr><td>Type of Winding </td><td>"+tw+"</td><td>
</td></tr>");
out.println("<tr><td>No. of Parallel paths </td><td>"+a+"</td><td>
</td></tr>");
out.println("<tr><td>Current per parallel path
</td><td>"+ipp+"</td><td>amps </td></tr>");
out.println("<tr><td>Flux per pole </td><td>"+fp1+"</td><td>
Wb</td></tr>");
out.println("<tr><td>No of slots </td><td>"+s+"</td><td></td></tr>");
out.println("<tr><td>No of conductors </td><td>"+z1+"</td><td>
</td></tr>");
out.println("<tr><td>No of coils </td><td>"+c+"</td><td> </td></tr>");
out.println("<tr><td>Armature conductors per slot</td><td>"+zs+"</td><td>
</td></tr>");
out.println("<tr><td>Slot Pitch </td><td>"+ys+"</td><td> </td></tr>");
out.println("<tr><td>Current Density of armature winding </td><td> 6
</td><td>A/mm<sup>2</sup> </td></tr>");
out.println("<tr><td>Area of Armature conductors
</td><td>"+aac+"</td><td> m<sup>2</sup></td></tr>");
out.println("<tr><td>Conductor Dimension </td><td>"+(aac/10)+" x 10
"+"</td><td> mm</td></tr>");
out.println("<tr><td>Slot Width </td><td>"+ws+"</td><td>m </td></tr>");
out.println("<tr><td>Slot Deapth</td><td>"+ds+"</td><td> m</td></tr>");
out.println("<tr><td> Armature mmf per pole </td><td>"+ata+"</td><td>
A</td></tr>");
out.println("<tr><td>mmf for Air Gap</td><td>"+atg+"</td><td>A
</td></tr>");
out.println("<tr><td>Gap contraction Factor</td><td>"+" 1.15 "+"</td><td>
</td></tr>");
out.println("<tr><td>Length of Air Gap</td><td>"+lg+"</td><td>
m</td></tr>");
out.println("<tr><td>Flux in Armature Core</td><td>"+fac+"</td><td>
Wb</td></tr>");
```

```
out.println("<tr><td>Flux Density in Core</td><td>"+" 1.2 "+"</td><td>Wb/
m<sup>2</sup></td></tr>");
}
//-------------------------------------------------------------------------------
----------
public void success3( )
{
out.println("<tr><td>Deapth of Armature core</td><td>"+dc+"</td><td>m
</td></tr>");
out.println("<tr><td>Internal Diameter of
Armature</td><td>"+di+"</td><td> m</td></tr>");
out.println("<tr><td>Leakage Co-efficient</td><td>"+"
1.15"+"</td><td></td></tr>");
out.println("<tr><td>Flux in the pole body</td><td>"+fpp+"</td><td>Wb
</td></tr>");
out.println("<tr><td>Flux density in the pole</td><td>"+" 1.5
"+"</td><td>Wb/ m<sup>2</sup></td></tr>");
out.println("<tr><td>Area of pole body</td><td>"+ap+"</td><td>
m<sup>2</sup></td></tr>");
out.println("<tr><td>Axial length of pole</td><td>"+lp+"</td><td>
m</td></tr>");
out.println("<tr><td>Nett Iron Length of pole
body</td><td>"+lpi+"</td><td> m</td></tr>");
out.println("<tr><td>Width of pole body</td><td>"+wp+"</td><td>
m</td></tr>");
out.println("<tr><td>Field mmf at full
load</td><td>"+atfl+"</td><td>A</td></tr>");
out.println("<tr><td>Space Factor</td><td>"+sf+"</td><td> </td></tr>");
out.println("<tr><td>Depth of field winding</td><td>"+" 45 "+"</td><td>
m</td></tr>");
out.println("<tr><td>mmf per metre of winding
</td><td>"+atwh+"</td><td>A/ m</td></tr>");
out.println("<tr><td>Height of field winding</td><td>"+hf+"</td><td>
m</td></tr>");
out.println("<tr><td>Height of pole shoe</td><td>"+" 20
"+"</td><td>mm</td></tr>");
out.println("<tr><td>Field Lamination thickness</td><td>"+" 10
"+"</td><td>mm</td></tr>");
out.println("<tr><td>Pole Height</td><td>"+hp+"</td><td> m</td></tr>");
out.println("<tr><td>Flux in Yoke</td><td>"+fl+"</td><td>Wb</td></tr>");
out.println("<tr><td>Area of Yoke</td><td>"+ay+"</td><td>
m<sup>2</sup></td></tr>");
```

```
out.println("<tr><td>Depth of Yoke</td><td>"+dy+"</td><td>
m</td></tr>");
out.println("<tr><td>Outer Diameter of Yoke</td><td>"+ddy+"</td><td>
m</td></tr>");
out.println("<tr><td>Flux Density in Yoke</td><td>"+" 1.5
"+"</td><td>Wb/ m<sup>2</sup></td></tr>");
out.println("<tr><td>Flux Density of Pole</td><td>"+bg+"</td><td>Wb/
m<sup>2</sup></td></tr>");
out.println("<tr><td>Duct Gap Contraction
Factor</td><td>"+kgd+"</td><td></td></tr>");
out.println("<tr><td>Slot Gap Contraction
Factor</td><td>"+kgs+"</td><td></td></tr>");
out.println("<tr><td>Gap Contraction Factor</td><td>"+kg+"</td><td>
</td></tr>");
out.println("<tr><td>Mmf for teeth</td><td>"+att+"</td><td>
A</td></tr>");
out.println("<tr><td>mmf for core</td><td>"+atc+"</td><td>A</td></tr>");
out.println("<tr><td>mmf for
Yoke</td><td>"+aty+"</td><td>A</td></tr>");
out.println("<tr><td>Total Field mmf</td><td>"+attt+"</td><td>
A</td></tr>");
out.println("<tr><td>Total Full Load mmf</td><td>"+atttfl+"</td><td>
A</td></tr>");
out.println("<tr><td>Voltage Across each Field
Winding</td><td>"+ef+"</td><td>volts</td></tr>");
out.println("<tr><td>Resistivity of Copper</td><td>"+" 0.021
"+"</td><td>ohm/m/mm<sup>2</sup> </td></tr>");
out.println("<tr><td>Length of mean turn</td><td>"+lmt+"</td><td>
m</td></tr>");
out.println("<tr><td>Area of conductor</td><td>"+af+"</td><td>
m<sup>2</sup></td></tr>");
out.println("<tr><td>No of turns</td><td>"+tf+"</td><td></td></tr>");
out.println("<tr><td>Resistance of each coil</td><td>"+rf+"</td><td>ohm
</td></tr>");
out.println("<tr><td>Field Current</td><td>"+If+"</td><td>amps
</td></tr>");
out.println("<tr><td>Field mmf provided</td><td>"+atflp+"</td><td>A
</td></tr>");
}
//-------------------------------------------------------------------------------
----------
}
```

6.7 INPUT DATA PAGE



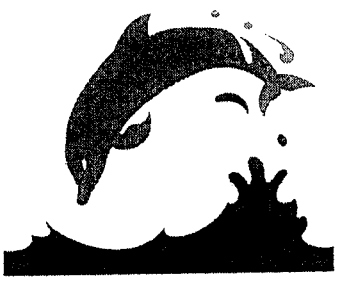
Results for your Data submitted:

6.8 Design Sheet

PARAMETERS	VALUES	Units
Voltage Rating	240	Volts
Power Rating	75	KW
Speed	1000	rpm
No of Poles	4	
Flux Density	0.71	Wb/m2
Average Flux Density	0.5	Wb/m2
Ampere Conductors	35000	A/m
Power generated	77.2	(KW)
Synchronous Speed	16.67	rps
Output Co-efficient	172.7	
Armature Diameter	0.400	m
Core Length	0.17	m
Pole Pitch	0.314	m
Pole Arc	0.210	m
Peripheral Speed	21	m/s
Frequency	33.34	Hz
Nett Iron Length	0.144	m
Line Current	340	Amps
Field Current	2.76	Amps
Internal Voltage Drop	11.3	volts
Armature Current	336	amps
Generated EMF	228.7	volts
No. of Parallel paths	2	
Current per parallel path	168	amps
Flux per pole	0.0254	Wb
No of slots	45	
No of conductors	270	
No of coils	135	

Armature conductors per slot	6	
Slot Pitch	0.028	
Current Density of armature winding	6	A/mm2
Area of Armature conductors	0.0000275	m2
Conductor Dimension	28 x 10	mm
Slot Width	0.013	m
Slot Deapth	0.03	m
Armature mmf per pole	5690	A
mmf for Air Gap	3450	A
Gap contraction Factor	1.15	
Length of Air Gap	0.005	m
Flux in Armature Core	0.0127	Wb
Flux Density in Core	1.2	Wb/ m2
Depth of Armature core	0.075	m
Internal Diameter of Armature	0.19	m
Leakage Co-efficient	1.15	
Flux in the pole body	0.0292	Wb
Flux density in the pole	1.5	Wb/ m2
Area of pole body	19.47	m2
Axial length of pole	0.17	m
Nett Iron Length of pole body	0.144	m
Width of pole body	0.127	m
Field mmf at full load	5060	A
Space Factor	0.676	
Depth of field winding	45	m
mmf per metre of winding	46300	A/ m
Height of field winding	0.110	m
Height of pole shoe	20	mm
Field Lamination thickness	10	mm
Pole Height	0.14	m
Flux in Yoke	0.0127	Wb
Area of Yoke	0.00847	m2
Depth of Yoke	0.055	m
Outer Diameter of Yoke	0.8	m
Flux Density in Yoke	1.5	Wb/ m2

Flux Density of Pole	1.5	Wb/ m2
Duct Gap Contraction Factor	1.018	
Slot Gap Contraction Factor	1.194	
Gap Contraction Factor	1.215	
Mmf for teeth	480	A
mmf for core	30	A
mmf for Yoke	300	A
Total Field mmf	4400	A
Total Full Load mmf	5060	A
Voltage Across each Field Winding	48	volts
Resistivity of Copper	0.021	ohm/m/mm2
Length of mean turn	0.774	m
Area of conductor	0.00000177	m2
No of turns	1890	
Resistance of each coil	17.4	ohm
Field Current	2.76	amps
Field mmf provided	5210	A



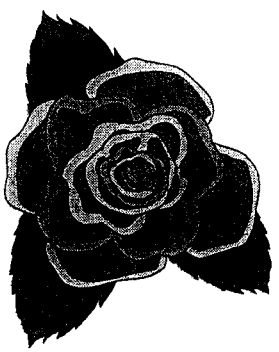
CONCLUSIONS

CHAPTER 7

CONCLUSION

A program in Java has been developed and tested. The application of this program for design of DC machine has been made. The design procedure is available on the web site. It can be accessed by anyone through internet and avail the facilities provided in the web site.

Some of the special features of this web application is the availability of a chat center and real time on-line discussions. Also several other modules such as EEE jobs and trends are added. All provisions are been made to make the application compatible for future developments.



REFERENCE

REFERENCES

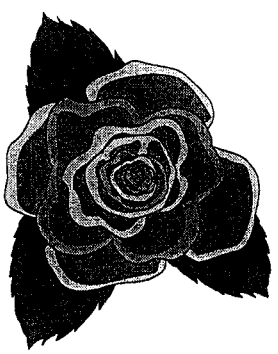
1. Michael Morrison,et al, "JAVA UNLEASHED " Techmedia NewDelhi, 1998

2. Andrew S Tanenbaum "Computer Reference", Tata McGraw-Hill Publications,1998

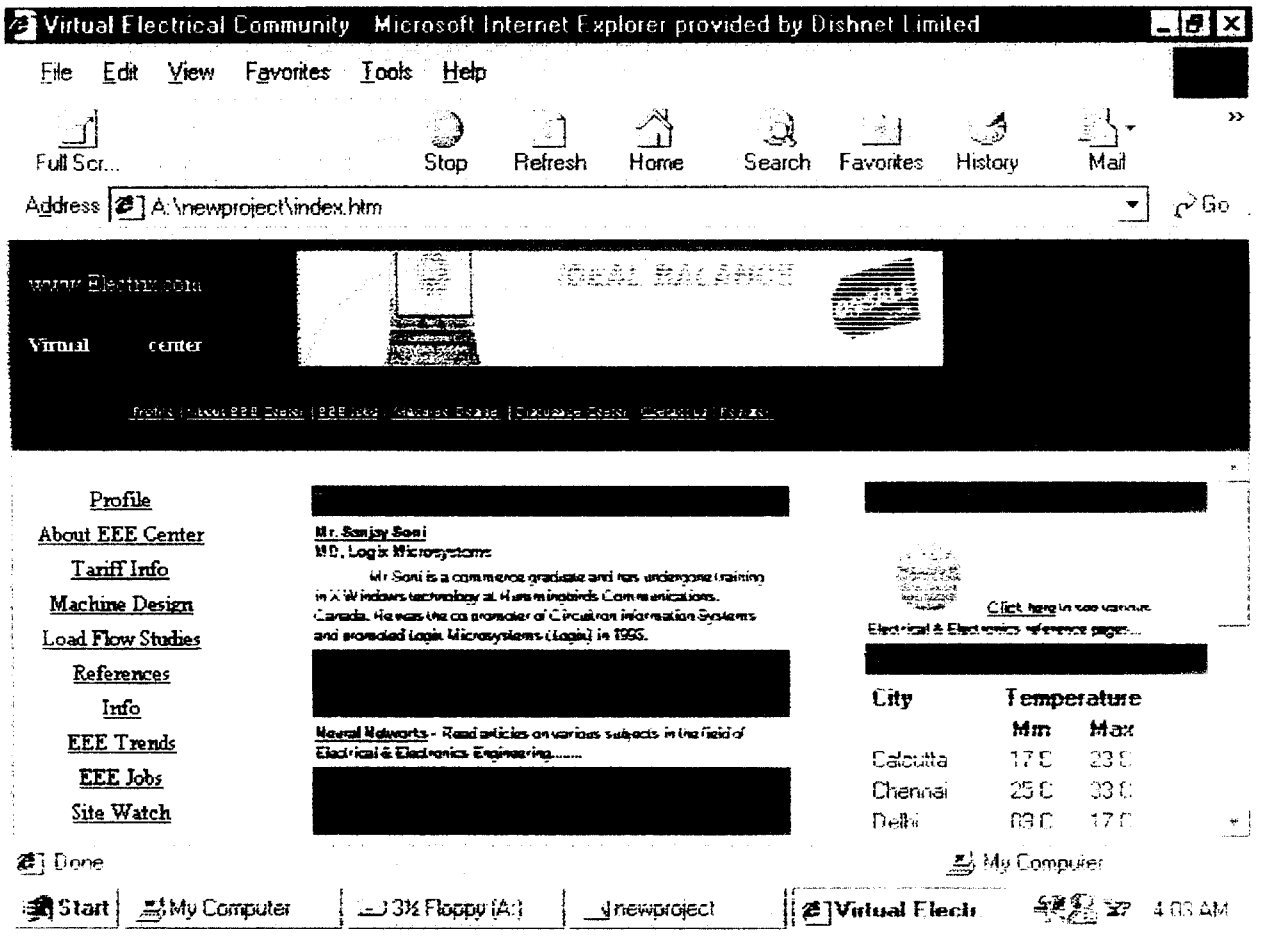
3. E BALAGURUSAMY, " Programming with JAVA ", Tata McGraw-Hill Publications,1999

4. A.K Sawhney, "Electrical Machine Design", Dhanpat Rai &Co, 1998

5. V.K Mehta, "Principles of Power System", S.Chand &Co Ltd, New Delhi, 1998.



APPENDIX


Virtual Electrical Community Microsoft Internet Explorer provided by Dishnet Limited
File Edit View Favorites Tools Help
Full Scr... Stop Refresh Home Search Favorites History Mail
Address A:\newproject\index.htm
Go
Virtual center
Profile
About EEE Center
Tariff Info
Machine Design
Load Flow Studies
References
Info
EEE Trends
EEE Jobs
Site Watch
Mr. Sanjay Soni
MD, Logix Microsystems
Mr Soni is a commerce graduate and has undergone training in X Windows technology at Hummingbirds Communications, Canada. He was the co promoter of Circuitron Information Systems and promoted Logix Microsystems (Logix) in 1995.
Neural Networks - Read articles on various subjects in the field of Electrical & Electronics Engineering.....
Click here to see various Electrical & Electronics reference pages...
City
Temperature
Min
Max
Calcutta 17 C 33 C
Chennai 25 C 33 C
Delhi 09 C 17 C
Done
My Computer
Start
My Computer
3½ Floppy (A:)
newproject
Virtual Electr
4:03 AM

Registration Form

Kindly enter this form & send it to us & get free newsletters & updations through your mail.

Name:

sex : (•) Male ( ) Female

Date of Birth : (mm/dd/yyyy)

Status : Employed

Organization :

Address 1:

Address 2 :

City :

State:

Country :

E-mail :

Phone :

Your comments about our service :

Send this Form Reset