

A. *Title*

Client/Server Technology

B. *Introduction*

Client/server is a network architecture that divides functions into client and server subsystems, with standard communication methods to facilitate the sharing of information between them. A client is defined as a requester of services and a server is defined as the provider of services. Client/server computing [2] is a very efficient and safe means of sharing database information in a multi-user environment. The client program usually accepts user requests and provides screen displays. Server programs generally reside on more powerful machines and are used to process information. A single machine can be both a client and a server depending on the software configuration. When client-server programs are on different machines, the client and the server machines are linked together through a network.

A common client/server example [5] is a print server. One can attach a printer to a PC and then share it with other users across the network. It serves print requests from all users, and off loads these requests from local machines. Another example is a mail server which functions much like a post office, receiving mail centrally and delivering messages to individual clients.

Some of the most popular applications on the Internet [4] follow the client/server design. For example: Email clients, FTP (File transfer) clients and Web browsers. Each of these programs presents a user interface (either graphic- or text-based) in a client process, that allows the user to connect to servers. In case of email and FTP, the user enters a computer name (or IP Address) into the interface to set up future connections to the server process.

In a client/server environment [3], most of the data is processed on the server and only the result are returned to the client. This reduces the amount of network traffic between the server and the client machine, which improves network performance. A central computer which is the server [2] responds to requests made by workstations which is the client for information or resources. The server not only controls the distribution of the information, but also engages in optimizing resource utilization. When the client completes editing the information, the server receives the changed information and applies the changes on the original.

The main advantages of client/server technology [3] are high performance and scalability. Thousands of users can access the same database at the same time, and the database can store billions of records.

Client-server applications [6] are modularized into client, server, and networking components. The resources used by a client-server application can also dictate the components, and thus the structure, of the client-server application. These resources include the number of server machines involved in the application, the location of the database(s), and the number of databases needed by the application.

The idea of splitting an application along client/server lines [9] has been used over years to create various forms of Local Area Network (LAN) software solutions. These solutions, however, are distinguished by the nature of the service they provide to their clients. For example With a file server, the client passes request for file records over a network to the file server. With a database server, the client passes Structured Query Language (SQL) requests as messages to the database server. With a transaction Server, the client invokes remote procedures that reside on the server with an SQL database engine. And with a web server, the

entire exchange of data is mediated by the browser and server using HTTP, this new model of client/server consists of thin, portable, universal clients that talk to super fat servers.

The performance of a client-server application [6] is affected by the client-server design structure of the application. The use of the appropriate client-server design structure combined with application logic design will yield the optimal performance for the application. The components of the client-server application force the modularity of the application code. In an application code, modularity will allow ease of updates to adopt to new environments. The key to application modulation lies in the resources needed by the application. The resource requirements of an application defines five design structures: Two-tier, Three-tier, Distributed , Parallel processing and Hybrid.

Middleware is a key to developing three-tier client/server application. Middleware is referred [12] to as the glue that holds together client/server applications. Middleware covers all the distributed [9] software needed to support interactions between clients and servers. Database-oriented middleware provides [12] an Application Program Interface (API) access to a database. Middleware enables developers to easily link an application to popular databases. Java Database Connectivity (JDBC) is a familiar API, these classes can be used to help an applet or a servlet access any number of databases without understanding the native features of the database.

The design structure of each application must be evaluated individually to take advantage of differences in capability, development effort, development time, degree of flexibility, and performance.

C. Literature Review And Current State-Of-The-Art

In a typical mainframe architecture [7] all processes are within the central host computer. Users interact with the host through a terminal that captures the user input and sends that information to the host. User interaction is through PCs and UNIX workstations. Though this architecture is effective, it is expensive, difficult to maintain and does not easily support graphical user interfaces or access to multiple databases from geographically dispersed sites.

The earlier versions of client/server were based on file sharing architectures [8], where the server downloads files from the shared location to the desktop environment. The requested user job is then run on the desktop environment and when it is finished it is returned to the server. File sharing architectures work if shared usage is low, update contention is low, and the volume of data to be transferred is low.

Due to the limitations of file sharing architectures [7], the client/server architecture emerged. The functionality of the application was separated logically into two parts, one being the processes requiring the majority of the computing they were put on the server, second the user interface and less processor-intensive processes, they were put on the client. This approach introduced a database server to replace the file server, using a relational database management system (DBMS). The client/server architecture reduces network traffic by providing a query response rather than total file transfer. It also improves multi-user update mechanism through a GUI front end to a shared database.

In a two-tier client/server architecture [10], the client communicates directly with the database server. The application or business logic either resides on the client or on the database server in the form of Structured Query Language (SQL) or stored procedures. A two-tier client/server design first came into seen with the applications developed for local

area networks in the late 1980's and early 1990's, and was mostly based on simple file sharing methods implemented by X-base style products (dBase, FoxPro, Clipper, Paradox, etc).

The Graphical User Interface (GUI) [10] became very popular in desktop environment. The general purpose LAN file server was replaced by a specialized database server. This model brought about the emergence of new development tools like PowerBuilder, Visual Basic, and Delphi. As the application becomes more complex the client gets fatter and the client needs a very powerful hardware to support it. In addition, the network using fat clients is huge, so that the effective bandwidth of the network, and thus the corresponding number of users who can effectively use the network, is reduced.

An alternative to fat Client [10], is thin client and fat server configuration. Here the user invokes procedures stored at the database server, this is another approach which is used in the two-tiered architecture. The fat server model has a better performance than fat client approach because the effective use of the network. The disadvantage of this approach is that stored procedures require proprietary customization and coding as they rely on a single vendor's procedural functionality.

In two-tier client/server systems [9] the application logic is either buried inside the user interface on the client side or within the database on the server side (or both). In three-tier client/server systems, the application logic (or process) lives in the middle-tier, it is separated from the data and user interface.

Client/server technology is slowly advancing [11] from two-tier architectures to three-tier architectures for many reasons. The most important being that increase in the number of users results in deterioration of performance of two-tier architecture. The second reason is

that the implementation of the processing management services provide little flexibility and limited choice of database management systems for the application.

Many of these limitations [11] in two-tier client/server architecture have been overcome by the three-tier architecture through the use of a middle tier. The middle tier works between the user interface which is the client and the data management which is the server and provides process management. Using a three-tier system results in improvements in flexibility, maintainability, reusability, and scalability. It enhances these measures by centralizing process logic. This makes administration and change management easier by localizing system functionality.

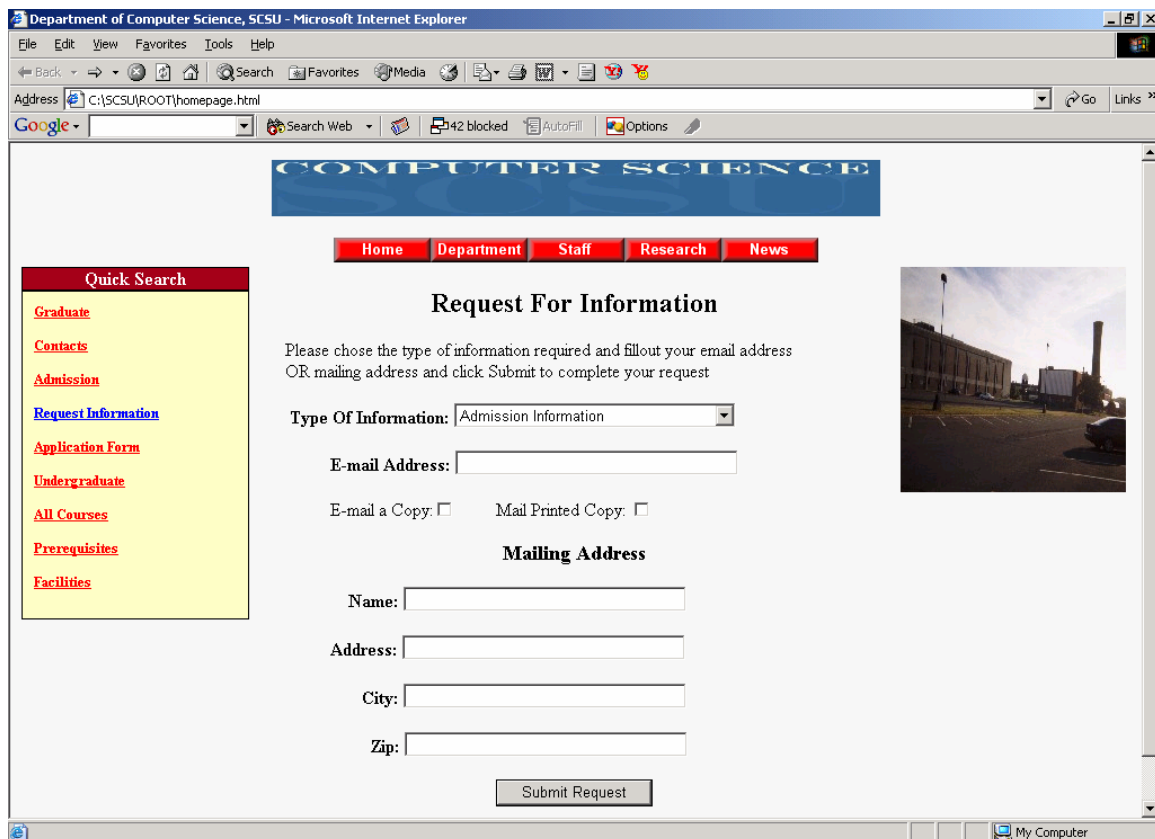
The goal of this project is to develop and implement a user-friendly web based client server application to handle user traffic to the departmental website. It will allow the visitors to the departmental web site to request information and will save their information on an oracle database and will create summary reports. The application will be built in Java platform using Oracle Database.

D. Methodology

This project involves the design and development of a three-tier client/server application. The application will allow all the visitors to the computer science departmental web site to request and obtain different kinds of information related to the department. The request is then saved on a relational database for further processing and analysis. This application will create several summary reports.

This application allows the user to:

- (1) Register with their username, password and email address. The information will then be stored in a database and helps the users to log on to the website to get information in the future.
- (2) Log on to the website using their user name and password and navigate through the website to get information about the computer science department. The login page is shown in Figure 1.
- (3) Request for printed material to be mailed to them or get the information through email to view the following detailed information. The request page is shown in Figure 2.



The screenshot shows a web browser window titled "Department of Computer Science, SCSU - Microsoft Internet Explorer". The address bar shows "C:\SCSU\ROOT\homepage.html". The page features a navigation menu with "Home", "Department", "Staff", "Research", and "News". A "Quick Search" sidebar on the left lists various links like "Graduate", "Contacts", "Admission", "Request Information", "Application Form", "Undergraduate", "All Courses", "Prerequisites", and "Facilities". The main content area is titled "Request For Information" and contains a form with the following fields and options:

- Type Of Information:** A dropdown menu currently set to "Admission Information".
- E-mail Address:** A text input field.
- E-mail a Copy:**
- Mail Printed Copy:**
- Mailing Address:** A section with four text input fields for "Name:", "Address:", "City:", and "Zip:".
- Submit Request:** A button at the bottom of the form.

A photograph of a building is visible on the right side of the page.

Figure2. User Request Page for printed material to be mailed or emailed.

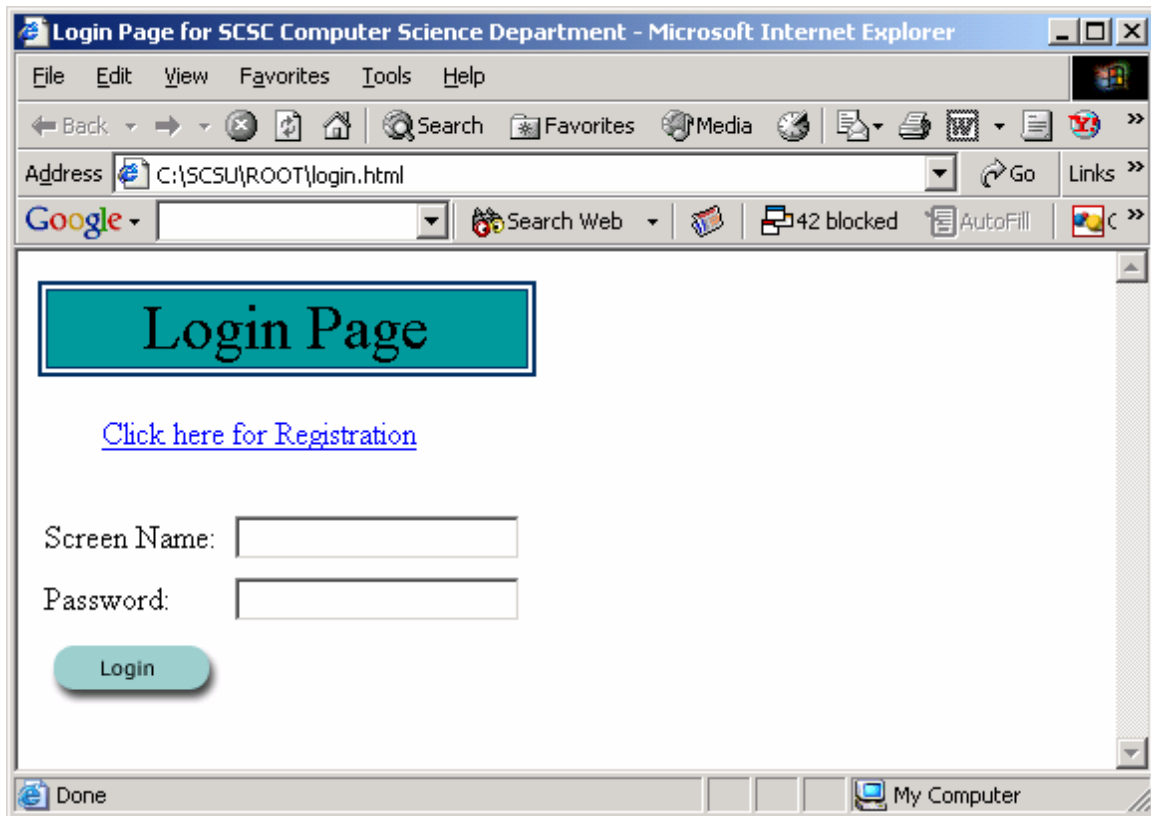


Figure 1. Login Page that allows the user to login to the website

- i. Admission Information such as eligibility requirements, also admission requirements for international students, which may be different from local students and different documents needed to complete the admission.
- ii. Course Information will contain the information on the available courses, outline of the course and other course-related information such as prerequisites, course description and books used.
- iii. Catalog: There are different catalogs to chose from such as Graduate Program offering, Graduate Calendar, Application and Admission, Tuition and Financial.

iv. Application Form: The user can complete the application form online when the appropriate link is sent to them or they can complete their application by using the printed form.

(4) A link to the International student information page that will contain information about cost, eligibility test, student visa, check lists etc

Following are some of the information available to the user about the department:

- (1) Information regarding faculty's email address, office hours, phone number and their office location in the school.
- (2) Departmental contact information such as email address, phone number, fax number, contact person and office location.
- (3) Information about location and direction to the computer science department.
- (4) The news page containing important news about the department and details regarding new courses that are being offered.

The information obtained will be stored in a relational database. Following are the types of information stored:

- (1) E-mail address: Email address is obtained during user registration. This will be stored along with the user name and password.
- (2) Mailing address: Mailing address is obtained when the user requests for printed material to be mailed to them.

- (3) Type of Request: User information request will be stored in the database. This will contain different types of information requested and a choice of how the user prefers to receive this information.

Faculty and other staff members use this information to generate reports containing a list of individuals who contacted the website. Reports will have a list of email and mailing addresses. This information may be used to send an e-mail regarding new course offerings to the people who contacted the website.

Figure 3 represents the detailed three-tier client/server application being developed

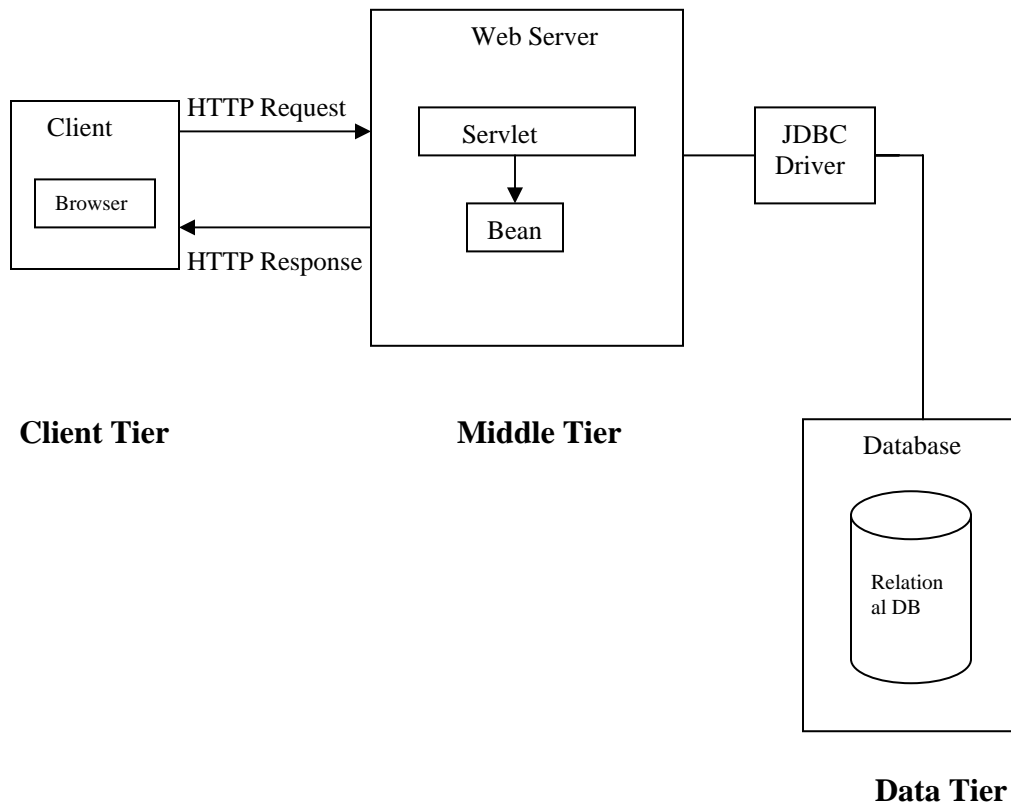


Figure 3. Detailed Three-tier client/server Architecture.

A typical three-tier client/server application consisting of three well-defined and separate processes, each running on a different platform. Figure.4 shows a three-tier architecture with three different platforms.

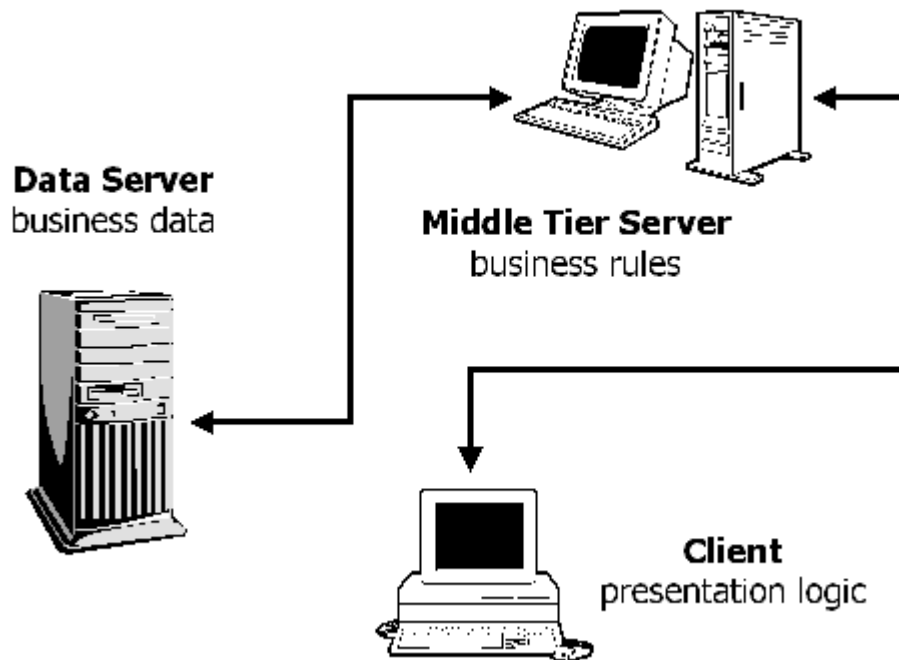


Figure4. Three-tier client/server Architecture [15]

Following are the three different platforms needed to run a three-tier application:

1. The client tier is a front-end (most often a PC) that communicates with the user through a graphical user interface. This is the direct interface between the user and the computer.
 2. The application server tier handles functions such as enforcing business rules, managing transactions and processing information. This is a functional module that actually processes data. This middle tier runs on a server and is often called the application server.
- The middle-tier server [18] plays a vital role in a three-tier application. It handles requests from clients by processing the request and returning the response to the client

and connecting to database, and retrieving information from the databases. Middle-tier servers provide business services to clients.

3. A database management system (DBMS) that stores the data required by the middle tier. This tier runs on a second server called the database server. The database tier provides data storage and other information services. The database tier consists of a standardized database that focuses on issues such as data integrity, security, redundancy, and shared services. Middle tier accesses the database tier with a driver supporting the JDBC.

Web application servers [13] provide a software framework to perform communications between the stateless HTTP browser and the stateful back-end application. Following are some of the typical program communication methods used by Web application servers: CGI (Common Gateway Interface), ASP (Active Server Pages), JSP (Java Server Pages), Applets, Servlets, ActiveX, CORBA (Common Object Request Broker Architecture), EJB (Enterprise Java Beans), RMI (Remote Method Invocation), XML (Extensible Markup Language) and XSL (Extensible Stylesheet Language).

The application is designed in the following segments:

- (1) Client: The client is the presentation layer. It presents data to the user in a user friendly format. It also sends and receives requests to and fro from one or more servers for processing data.
- (2) Web Server: A web server is a computer with special software to host web pages and web applications. Web server is a program running on the server machine, which accepts request from Client (Web browser) and sends back the result. Web server serves web pages to clients across the Internet or an Intranet. The web

server hosts the pages, scripts, programs, and multimedia files and serves them using Hyper Text Transfer Protocol (HTTP), a protocol designed to send files to web browsers and other protocols [16].

- (3) Servlets: When the server receives a request, it passes the request to the Servlet. The Servlet is an application program interface between the Web server and the application program. The program is loaded into the Web Server when the server starts up [14].
- (4) JDBC: Java Database Connectivity (JDBC) technology is an API that is used to access virtually any tabular data source from the Java programming language. It provides cross-DBMS connectivity to a wide range of relational databases. JDBC is used to connect the servlet program to database. The servlet queries the database through JDBC and converts the recordset returned by the database into an object [14].
- (5) Database: A relational database consists of a set of tables, where each table is a set of records. A record in turn is a set of fields and each field is a pair field-name/field-value. All records in a particular table have the same number of fields with the same field-names. Data are queried through JDBC from servlets and displayed on browser using user interface.

Documentation of this project will comprise a user's guide and detailed description of the different modules of the application. Complete listing of the program source code will be provided.

E. Contributions of the Project

Most Web applications work on client/server environment. These kind of applications are multi-tiered client/server applications which use different software and technology in each tier for developing an application. The purpose of this project is to outline a roadmap and guideline to client/server application development and will automate the information retrieval for users of computer science departmental web site.

F. References

1. Introduction to Client/server Networking

Web site: <http://compnetworking.about.com/library/weekly/aa050201a.html>

2. What is client/server computing?

Web site: <http://www.nutech.com.hk/solomon/clientserver.htm>

3. Client/Server Architecture

Web site: http://www.biblioscape.com/client_server.htm

4. Client/Server Applications

Web site: <http://compnetworking.about.com/library/weekly/aa050201a.htm>

5. Client/Server Examples

Web site: <http://www.darwinmag.com/read/090103/question11.html>

6. Client-Server Application Design Structure

Web site: <http://www.interex.org/pubcontent/interact/oct95/10wang/wang.html>

7. Client/Server Software Architectures--An Overview

Web site: http://www.sei.cmu.edu/str/descriptions/clientserver_body.html

8. The Evolution of Clients and Servers

Web site: <http://www.darwinmag.com/read/090103/question11.html>

9. Client/Server Survival Guide Second Edition By Robert Orfali, dan Harkey and
Jeri Edwards (Ch 1-5)
10. Client/Server and the N-Tier Model of Distributed Computing
Web site: <http://n-tier.com/articles/csovervw.html>
11. Client/server Research Findings
Web site: <http://homepages.wmich.edu/~s9chrism/Portfolio/Clientserver.html>
12. Modern Database System Sixth Edition By Jeffrey Hoffer Page 334
13. Overview of Web Application Servers
Web site: <http://developer.novell.com/research/appnotes/2000/january/06/02.htm>
14. Java Servlets Second Edition. Karl Moss. Servlets (Ch 1, 2, 8 & 9)
15. Architecture Diagrams
Web site: <http://www.linuxjournal.com/article.php?sid=3508>
16. Modern Database Management Sixth Edition. Jeffery A.Hoffer, Mary
B.Prescott, Fred R.Mc Padden page 363 – 388
17. Evolution of server architectures.
Web site: <http://www.ciol.com/content/search/showarticle1.asp?artid=35576>
18. Java Server Page Fundamentals.
Web site: <http://java.sun.com/developer/onlineTraining/JSPIntro/contents.html>
19. Java Server Pages [tm] Technology - White Paper.
Web site: <http://java.sun.com/products/jsp/whitepaper.html>