2001

Web-based database management system for research and development laboratories: Technical service support system

Benito Solórzano

WEB-BASED DATABASE MANAGEMENT SYSTEM FOR RESEARCH AND DEVELOPMENT LABORATORIES: TECHNICAL SERVICE SUPPORT SYSTEM

A Project

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Benito Solórzano

December 2001

WEB-BASED DATABASE MANAGEMENT SYSTEM FOR RESEARCH AND DEVELOPMENT LABORATORIES: TECHNICAL SERVICE SUPPORT SYSTEM

A Project

Presented to the

Faculty of

California State University,

San Bernardino

---

by

Benito Solórzano

December 2001

Approved by:

[redacted]

Josephine G. Mendoza, Chair,
Computer Science

11-26-01
Date

[redacted]

George Georgiou

[redacted]

Ernesto Gomez

ABSTRACT

With the use of the Internet and the emerging of e-commerce, new and improved technologies and modeling techniques have been used to design and implement web-based database management systems.

This project proposes the design and implementation of a web-based database management system for research and development laboratories, using object-oriented-modeling techniques.

Requirements collection and analysis were performed using an EER (Entity Enhanced Relational) model since it is adequate to capture, in a simple and general model, all the database requirements. This EER model was transformed to an object-oriented model for further analysis. Since an object-oriented model is precise and concise it was used to implement the web-base database using Oracle8i, an object-relational database management system.

The importance of the project resides in the fact that database management systems are adequate to manage (store, retrieve and control) data in research and development laboratories. Data has to be stored in an efficient manner so that -(1) research and development of new products can be conducted properly and (2) customer and manufacturing issues can be solved in a timely manner.

The database management system is accessed through the internet using a web browser. The interface used between the internet and the database was built using JSP (Java Server Pages). JDBC (Java Database Connectivity) and SQLJ (Structured Query Language embedded in Java Code) were used as database access schemas. These two database access schemas are complementary approaches for a web-based database management system that uses static SQL statements which are used in this project. The SQLJ Query and JDBC Query JSPS have a drawback that Java logic is embedded within HTML code and it may not be adequate for complex database operations. Thus it makes sense to separate the logic for generation of dynamic content from its presentation, using session scoped Java Beans.

This project can serve as a prototype to design, analyze and implement a web-based database management system for a research and development laboratory since this type of laboratories performs similar activities such as research and development of new product, customer service and manufacturing support.

Also, the modeling techniques and technologies that have been used in this project are adequate and flexible enough to capture special requirements that could be out of the scope of this project.

ACKNOWLEDGMENTS

Quiero agradecer primeramente a Jesus mi fiel amigo, gracias Señor , muchas gracias. Dedico en especial este trabajo a mis padres Gilberto Solórzano y Rosa Rubio de Solórzano por su apoyo incondicional en toda mi vida. Gracias papas por haberme enseñado y motivado todo el tiempo a seguir adelante y haber confiado todo el tiempo en mí dándome apoyo moral, espiritual y económico.

Dedico este trabajo con especial cariño a mis hermanos: Irma, Doralina, Pablo, Vicente, Angélica, Alma Rosa, Diana (gracias mi Dayana) y Jorge. Cada uno de ustedes siempre confió en mi, en que iba a lograr una carrera profesional y cada uno de ustedes me lo hicieron saber en diferentes maneras. Gracias a cada uno de ustedes que representa a cada una de sus familias a los cuales agradezco su ayuda y apoyo.

Lo dedico tambien a mi tía Teresita y a mi tía Angeles y a mis primos hermanos: Claudia, Miriam, Adriana, Moises, Cuautémoc, Mauricio, Alfonso, Lourdes e Isaí.

Lo dedico tambien a mi amiga Nitza Vazquez. Gracias Nitza por toda tu ayuda en todo tiempo.

Gracias a mi pastor Hno Eliseo Perez por todo su apoyo, con su ejemplo he aprendido a darle gracias a Dios en todo tiempo y definir un caracter cristiano, doy gracias a Dios por haberle conocido.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

SOFTWARE REQUIREMENTS AND SPECIFICATION

Introduction

Most of manufacturing companies have research and product development facilities to keep pace with the rapid changes in design and special materials needs in the industry. The main activities of these types of laboratories are: manufacturing support, research and development of new products, and technical support for customer issues. The main actors involved in these types of activities are: managers, laboratory technicians, engineers, technical service representatives and customers.

The main objective of this project is to develop an interactive system to provide database management so that these types of laboratories can respond in a timely manner to customer demands.

A web-based database management system should be in place to support these types of laboratories. This project proposes a system for database management to record and organize in an efficient manner all data generated by the three main areas of research and development facilities:

manufacturing, technical service, and research and development. The system should be able to record:

- All testing performed on manufactured products so that engineers and managers can use this information to solve any type of manufacturing issue.
- All testing performed on samples pertaining to customer issues; including process information and test results so that managers can give customers advise when similar issues come up.
- Descriptive data of any on-going project related to research and development of new products so that updated information of any new product in development is updated and available for future reference.
- All testing performed on samples generated by the projects so that decisions can be taken to conduct research in the proper direction.
- Descriptive data of company contacts that provide information related to the projects so this information and its source can be referred for future projects.
- Descriptive data of all raw materials vendors so that engineers can appropriately contact vendors

when behavior variation is noticed on raw materials.

- Data of all patents being generated by the projects so that records of all inventions made within the lab can be referred in the future.
- Descriptive data of manufacturing safety datasheets of raw materials and manufactured products so that engineers provide safety information related to raw materials or final products to workers or lab technicians in a timely manner.

The system should automatically generate regular monthly reports for all types of tests performed on manufactured products. The reports should be able to include statistical data e.g. average, standard deviation. This information is vital to take corrective action when manufacturing issues come up. The objective of this project is to design and implement a user-friendly web-based database management system to support technical service for customers in research and development laboratories of the circuit boards industry. This system can be used as a prototype to design and develop web-based database management systems for research and development facilities in different types of industries.

The database management system will be accessed through the internet using a web browser. The interface used between the internet and the database is built through Java Server Pages using Java Database Connectivity JDBC and SQLJ as database access schemas. The web site implementation will be done using JSP files.

The web-based database management system will have a browser-based interface (Java Server Pages) and JDBC and SQLJ will be responsible for passing data to the HTTP server (see Figure 1). [3]

This project proposes the system to be web-based so users (managers, engineers, tech service representatives and lab technicians) can access the database through the Internet and request, retrieve, add or remove data from it; according to the level access granted. This type of technology is of great advantage in manufacturing environments where database users are located in different locations and data have to be either shared or delivered in a timely manner. For instance, tech service representatives should be able both to request any type of testing related to a specific customer issue and get testing results back through the internet. Lab technicians should be able to access the main research and development database that contains the test results of all samples

generated from all previous projects so that Lab Technicians do not execute repetitive work and thus speed the process of the projects. Engineers and managers should be able to access a main manufacturing database that includes test data from all type of products and they will be able to share and work with the same information and avoid the situation where different people try to solve the same issue. Instead they could coordinate work as a team to solve any manufacturing issue.

In conclusion, the interactive system will help these types of laboratories to reduce administrative time and costs by avoiding duplicate work as in the case of projects testing. Once a new sample is generated by the research team, it will be tested and the data will be stored for future reference. The system will also help to streamline management process. There will be more time for managers to make decisions instead of taking that time to look for information or request testing to complete missing information. Work efficiency will increase due to a faster response to customers complaints and manufacturing issues.

Product Overview

The purpose of this project is to design and implement a web-based database management system for research and development laboratories that support manufacturing of circuit boards; including the three main areas that concern this type of laboratories: manufacturing, technical support and research and development of new products.

The scope of this project covers the design and implementation of the web-based database management system for the technical support area; and a user-friendly web-based database management system to support technical service for customers in research and development laboratories of the circuit boards industry.

There are several advantages in implementing this project using a database management system: [1] [2]

- Users will access a main database through the Internet so they can share information and work together to solve any type of issue.
- Technical service representatives will both request testing through the Internet and get test results back in a timely manner so that customers will get technical advice quickly.

- Users need to learn only one interface, that of the web browser. This has benefits in terms of reduced training costs and improved efficiency in manipulating data.
- The database management system will provide facilities for recovering from hardware or software failures. The backup and recovery subsystems of the DBMS ensure that the database is restored to the state it was in before the program started executing.
- A prime selling feature of the database approach is that developing a new application - such as retrieval of certain data from the database for printing a new report - takes very little time. Designing and implementing a new database from scratch may take more time than writing a single specialized file application. However, once a database is up and running, substantially less time is generally required to create new applications.
- The database management system will allow changes to the structure of the database without affecting the stored data and the existing application programs.

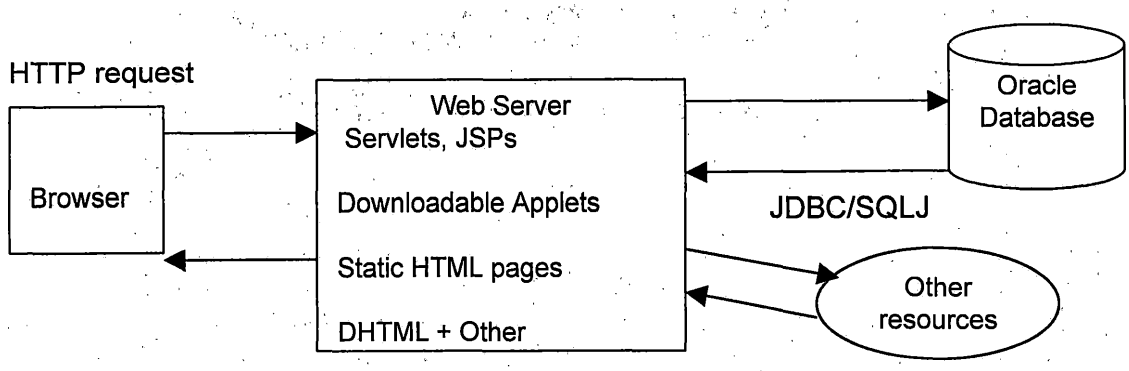
- The system will provide up-to-date information. As soon as one user's update is applied to the database, all other users can immediately see this update. This availability of up-to-date information is essential for manufacturing environments.
- The system will permit consolidation of data and applications, thus reducing the amount of wasteful overlap between activities of data-processing personnel in different projects or departments.
- The database system through its dictionary will define and enforce standards among database users in the organization. This facilitates communication and cooperation among various departments, projects, and users within the organization. Standards can be defined for names and formats of data elements, display formats, report structures, terminology and so on. The database administrator can enforce standards in a centralized database environment more easily than in an environment where each user group has control of its own files and software.

There are some drawbacks we have to keep in mind while developing the system:

- High initial investment in hardware, software and training since the web-based database management system will be implemented from scratch.
- Network response times are affected when the main database is accessed from different locations.

The database management system will be accessed through the Internet using a web browser. The interface used between the Internet and the database is done through Java Server Pages using Java Database Connectivity JDBC and SQLJ as Database access schemas and the Web site implementation will be done using HTML (See Figure 1).[12]

Figure 1. General Overview of the Interactive System



The main components are:

- Web server: Apache Server
- An http server (Server Application Program)
- Database connection schemas (JDBC and SQLJ)
- Database server (Oracle 8i)
- Jserv server to transform jsp files into servlets.

The Web browser acts as the front end and uses HTML forms for gathering user input. JDBC is responsible for passing data to the HTTP server and Java Server Pages function as an interface between the database and the browser. [3]

Product Perspective

System Interfaces

Netscape 4.7 and Internet Explorer 5.5 browsers are capable of instantiating a Java Virtual Machine that supports the JDK 1.1.8 API. Java Server Pages will be used to create a browser-based interface. [4][15]

User Interfaces

This project presents the design of a web-based database management system for research and development laboratories, but it only shows the implementation of the technical service support area since this part of the

project shows all thefeatures and technologies intended to be demonstrated in this project. However other areas can be implemented analogously since EER diagrams, object models and data dictionaries were created.

The four main elements of the web-based database management system for the technical service support area are: Home Page, Customer Menu, Technical Service Representative Menu, Laboratory Technician Menu and Manager Menu.

Home Page. The Home Page welcomes the user and ask for user id and password to authenticate the user. Depending on the type of user, the system shows four different menus: Customer Menu, Laboratory Technician Menu, Technical Service Representative Menu and Manager Menu.

Customer Menu. This menu allows the customer to select among four options: Add a request, retrieve test results, view historical data and exit see Appendix D.

Add a request. This menu allows the customer to submit a request to the research and development laboratory. The system asks the customer to enter all information pertaining to the sample to be tested. This menu gives 3 options: submit, reset and exit. When a test

request is submitted a request identification number is automatically assigned to that customer request.

Test results. Using this menu the customer can request for testing results from the system. The system asks the customer for the request id given to the customer test request. Once the request id is validated. If the test has been completed the customer gets a summary of test results. If the test has not been completed yet the customer gets a message saying to try again later.

Historical Data. This menu gives the customer the opportunity to get information on several previously submitted and completed test requests. The customer enters the selected date, the material type and the type of test, and selects yes or no on whether statistical summary is desired. If the selection was correct the system displays a table showing a summary of historical data with statistical information (average and standard deviation).

Exit. This menu allows the customer to return to the home page.

Technical Service Representative Menu. This menu allows the technical service representatives to select from four options: read customer requests, update test results, review historical data and exit. See Appendix E.

Read Customer Requests. When the technical service representative enters this menu, the screen shows a table containing all testing requests that have been submitted by different customers and which need to be reviewed. The table includes request id, customer name, date requested, sample identification, material type and brief description of the problem. The technical service representative can select any request that he wants to review first. Once a customer testing request has been selected the system shows a table including all information pertaining to the request such as: request id, customer's name, technical service representative name, sample id, material type, nature of the problem, lab support requested and sample process conditions. The customer has requested some laboratory testing but it is the technical service representative's job to (1) select the appropriate testing to be performed, (2) and give the laboratory technicians who will perform the testing some observations to take into account. When the technical service representative submits the final request, a message indicating that the final request has been submitted will be displayed.

Update Test Results. When the technical service representative enters this menu, the screen will show a table containing all customer-testing requests that have

been completed and that have not been seen by the technical service representative. The technical service representative can select the customer testing results that he wants to view first. Once a request id has been selected, a table showing the test results will be displayed. The technical service representative will be requested to enter the diagnosis and the technical advice. There are two selection buttons either to submit or to exit. When the technical service representative submits an input, a message saying that test results have been submitted will be displayed.

Historical Data. This menu allows the technical service representative to review historical data of a test request completed by the research and development laboratory in the past. The technical service representatives have access to information related to different customers. The window asks for the customer name, date (query will retrieve data from given date), material type, type of test, and indicate by yes or no, if statistical summary is desired. Once the technical service representative submits the request to view historical data, a table will be displayed that includes: request id, material type, sample id, nature of the problem, test results, diagnosis, technical advice, and observations.

Statistical information (average and standard deviation) is displayed if requested.

Exit. Allows the technical service representative to go back to the home page.

Laboratory Technician Menu. This menu presents four options: Read customer requests, report test results, read historical data and exit (see Appendix F).

Read Customer Requests. When the laboratory technician enters this menu, the system assigns automatically a testing request for the lab technician to perform. The lab technician is reminded to print the testing request using the browser.

Report Test Results. When the laboratory technician enters this menu, the screen will show a table containing all customer-testing requests that have been assigned to the technician for testing. The laboratory technician then reports test results on the selected testing request.

Once a request id has been selected the system displays the sample information for the request and asks the laboratory technician to enter the test results. There are two selection buttons either to submit or to exit. When the laboratory technician submits his input, a message saying that test results have been submitted will be displayed.

Historical Data. Same as Historical Data option in the technical service representative menu.

Exit. Allows the technical service representative to go back to the home page.

Manager Menu. This menu presents four options: Read customer request, update and release test results, read historical data and exit (see Appendix G).

Read Customer Requests. When the manager enters this menu, the system displays status of all requests submitted -- 'SUBMITTED', 'IN PROGRESS', 'TESTED', etc.

Update Test Results. When the manager enters this menu, the screen will show a table containing all customer-testing requests that have been completed and that have been seen by the technical service representative. The manager can select the customer testing results that he wants to update first. Once a request id has been selected, a table showing the test results will be displayed including diagnosis and the technical advice. The manager is asked to enter the final technical advice. There are two selection buttons either to submit or to exit. When the manager submits his input, a message saying that test results have been submitted will be displayed and the systems asks the manager to submit notification e-mail to the customer. This e-mail

lets the customer know that the request has been completed.

Historical data. Same as Historical Data option in the technical service representative menu.

Exit. Allows the technical service representative to go back to the home page.

Hardware Interfaces

Pentium II processor.

350 Mhz Level 2 Cache 512 KB.

Integrated System Memory, 64 MB SDRAM.

Video Memory 8 MB SGRAM.

Software Interfaces

Linux 6.2[13]

Apache 1.3 Web server.

JDK 1.1.8[4]

Oracle8i 8.1.7.0.1[17]

HTML 4.0[15]

Jserv Engine.

CHAPTER TWO

DATABASE DESIGN

The goals of database design are multiple: to satisfy the information content requirements of the specified users (lab technicians, tech service representatives, managers, engineers, and developers) and applications; to provide a natural and easy to understand structuring of information; and to support processing requirements and any performance objectives such as response time, processing time, and storage space.

Research and Development Laboratory Database

Before we can effectively design a database, we must know the expectations of the users and the intended uses of the database in as much detail as possible. The process of identifying and analyzing the intended uses is called requirements collection and analysis. Results of this design phase are expressed graphically using the EER (Enhanced Entity Relationship) model (See Figures 2, 3 & 4). [1][2]

Right Half

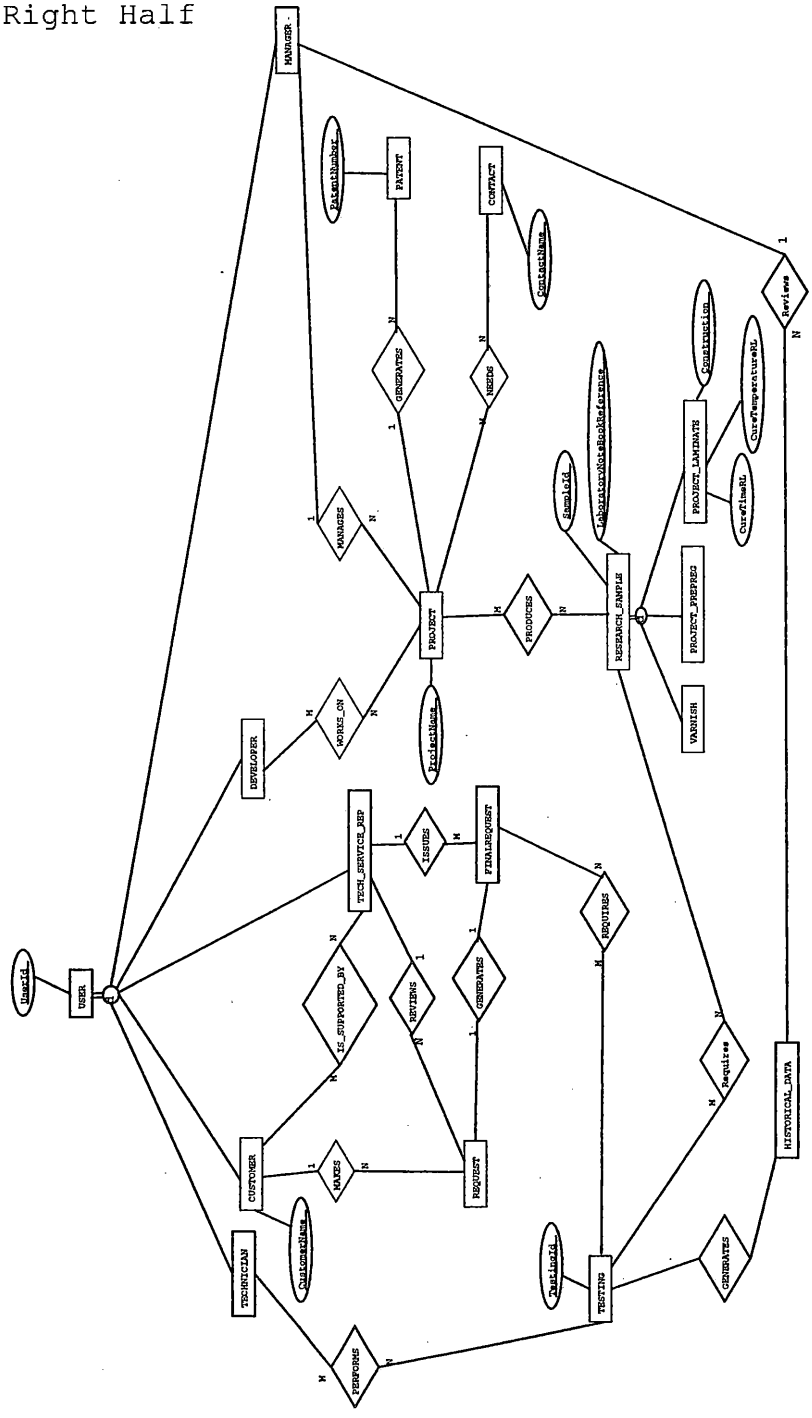
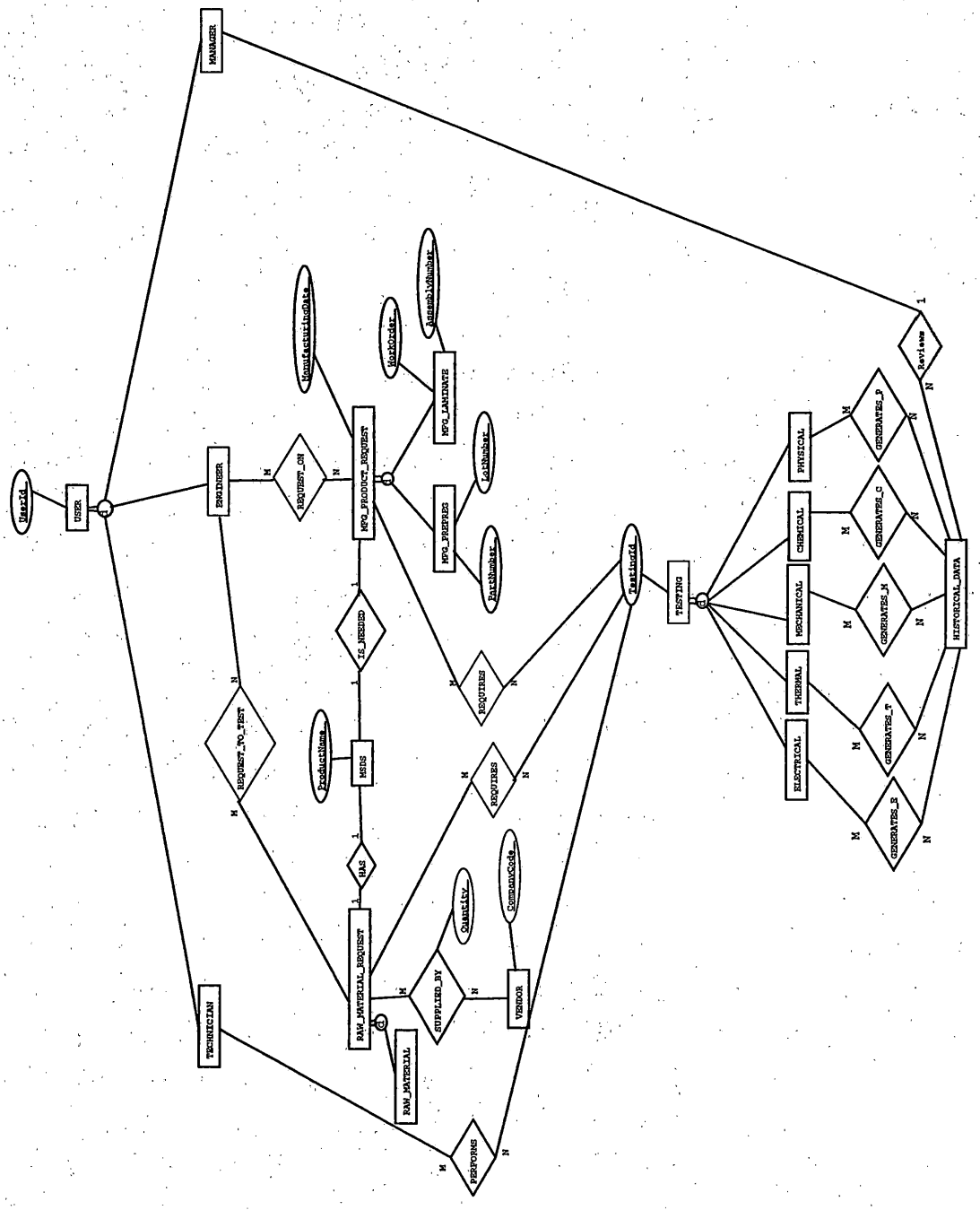


Figure 2. Entity Enhanced Relational Diagram: General Overview

Left Half


MANAGER
UserId
USER
ENGINEER
TECHNICIAN
REQUEST_ON
REQUEST_TO_TEST
MPQ_PRODUCT_REQUEST
ManufacturingDate
WorkOrder
AssemblyNumber
MPQ_LAMINATE
MPQ_PREPREG
LotNumber
PartNumber
IS_NEEDED
MSDS
ProductName
HAS
RAW_MATERIAL_REQUEST
RAW_MATERIAL
SUPPLIED_BY
Quantity
CommodityCode
VENDOR
REQUIRES
REQUIRES
PERFORMS
TestingId
TESTING
PHYSICAL
CHEMICAL
MECHANICAL
THERMAL
ELECTRICAL
GENERATES_P
GENERATES_C
GENERATES_M
GENERATES_T
GENERATES_E
HISTORICAL_DATA
Revises

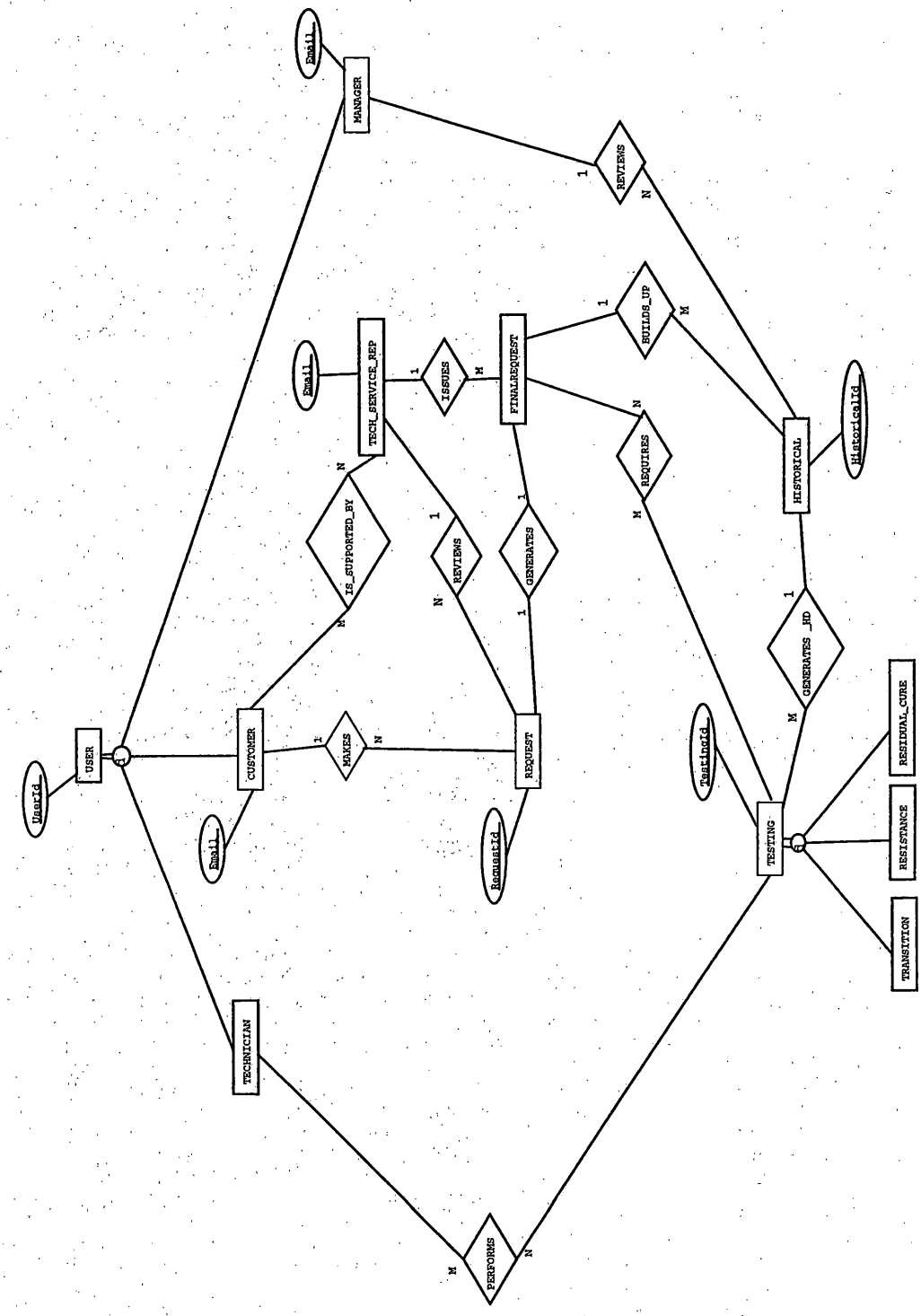


Figure 3. Entity Enhanced Relational Diagram: Technical Service Support

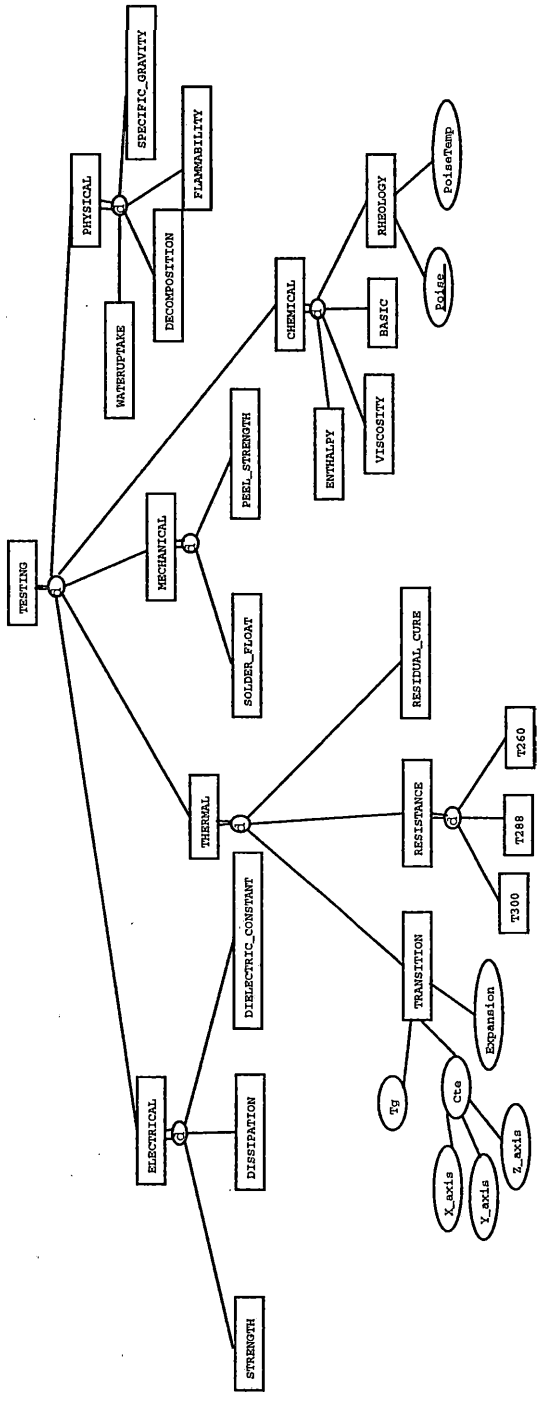


Figure 4. Entity Enhanced Relational Diagram: Testing Unit

The analysis and evaluation of the three main areas of research and development laboratories of the circuit boards industry are presented next.

The research and development of new products is done through projects. The manager assigns a specific project to a lab technician and each project generates samples to be tested for different types of chemical properties. [7] Therefore data has to be stored in an efficient manner since this information is used for on-going and future projects.

Data is currently stored under each project's folder and takes a lot of time to find needed information. It is sometimes easier to retest the samples than to look for the information in the manual filing system.

Technical support requests are issued to the lab by customers. The technical service representatives analyze the issue so that adequate testing can be requested to solve the problem. A customer submits his request and gets the testing results and technical advice back from the technical service representative by conventional means such as fax and mail. Since requests may come from different locations in the world the response time to requests is extremely high. All data generated by technical service requests is also stored in a

conventional filing system. This makes difficult the retrieval of specific test results for future similar problems. This problem causes to re-test 100% of testing requests since there is no way to retrieve in a timely manner information generated from previous requests and advice customers using historical data.

Research and development laboratories are also in charge of solving manufacturing issues using historical data. When a manufacturing issue comes up, a sample is taken from the process and it is tested for all the properties and these tests results are compared to historical data of the product (database) so that a course of action can be taken. Manufacturing historical data is currently stored using spreadsheets.

The analysis suggests that an efficient web-based database management system should be in place to support these types of laboratories. This project will develop a system for database management to record and organize in an efficient manner all data generated by the three main areas manufacturing, technical service and research and development. The system should be able to keep record of:

- Data for testing performed on manufactured products (prepreg and laminates), Data includes: Tg, CTE, thermal expansion (T-288 and T-260),

peel strength, specific gravity, electrical strength, water up-take, dielectric constant, dissipation factor, decomposition point, residual cure, solder float. [8]

- Data for testing performed on customer testing requests, data includes: Tg, CTE (x, y, and z), thermal expansion, T288-T260. The database includes descriptive data on the customer requesting the testing such as: customer contact, phone, fax, sales representative, technical service representative, material type, lots numbers affected, nature of customer problem, lab support being requested, customer process conditions related to the issue. [7][8][9]
- Descriptive data of any on-going project within the laboratory.
- All testing performed on varnishes, prepregs, and laminates generated by the projects for future retrievals since this information will be relevant for other projects.
- Descriptive data of company contacts that provide the information related to the projects, such as name salutation, job title, employer,

manager, subordinates, secretary, addresses (home, office, mailing), phone numbers (home, office, fax, secretary), e-mail addresses and personal comments.

- Descriptive data for patents being generated by the projects, including patent number, title, subject, and claims and descriptive data for vendors, such as company code, supplier, division, phone number, fax, e-mail, address (city, state, zip code, region, country).
- Descriptive data for MSDS (Manufacturing Safety Datasheet) of samples introduced into the laboratory and products manufactured within the plant, data such as: MSDS ID, product name, chemical name or composition, CAS, purpose, chemical family, evaluation, distributor, amount, unit, specific gravity, flash point, date received, emergency phone number.
- Descriptive data of all MSDS of raw products and manufactured products.

All requirements are represented in the EER diagrams (see Figures 5a-5c, 6a-6f, 7a-7e & 8a-8c). [2]

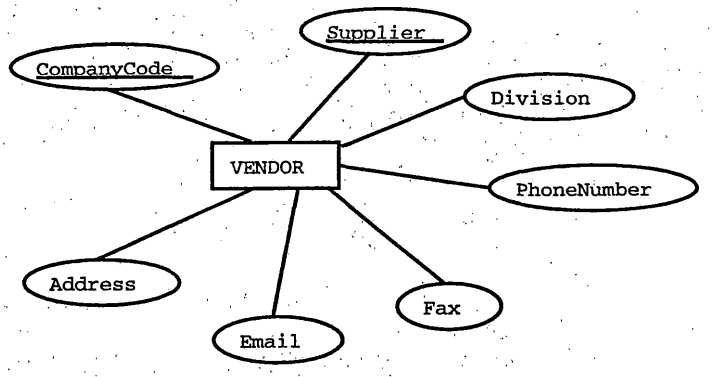


Figure 5a. Vendor Entity Properties

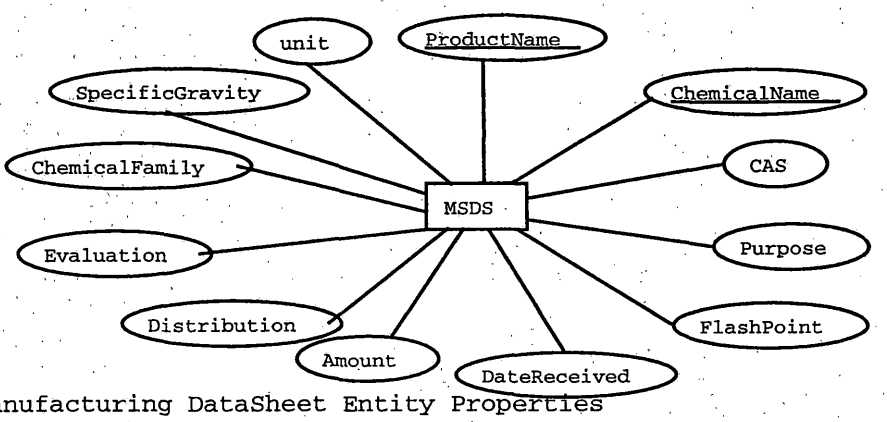


Figure 5b. Manufacturing DataSheet Entity Properties

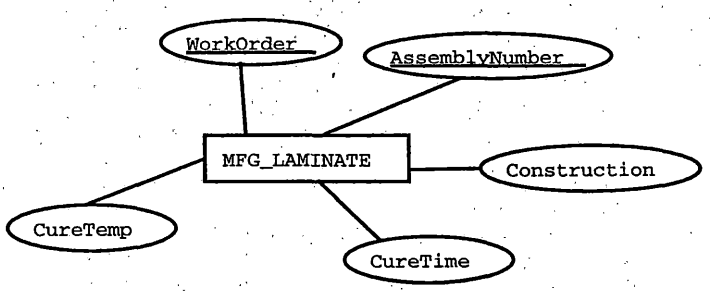


Figure 5c. Manufacturing Laminate Entity Properties

Figure 5. Attributes of Vendor, Manufacturing Safety Datasheet and Manufacturing Laminates Entities

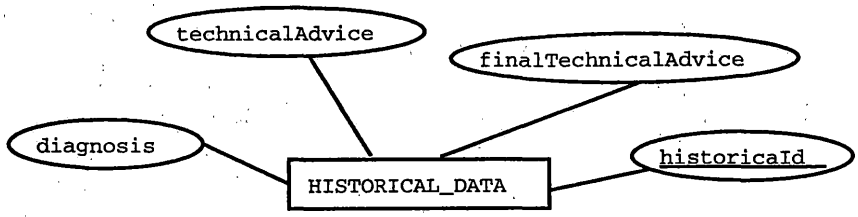


Figure 6a Historical Data Entity Properties

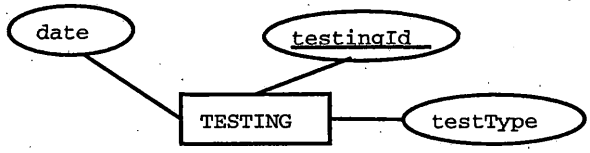


Figure 6b Testing Entity Properties

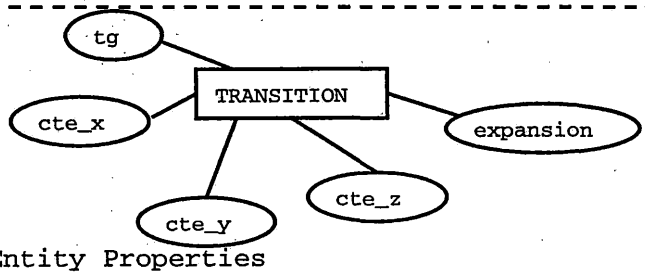


Figure 6c Transition Entity Properties

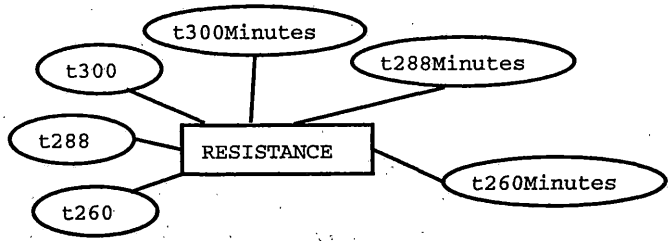


Figure 6d Resistance Entity Properties

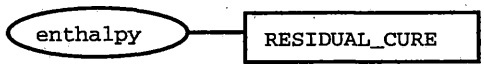


Figure 6e Residual Cure Entity Properties

Figure 6. Attributes of: Technical Service Laminate, Technical Service Prepreg, Historical Data, Testing, Transition, Resistance and Residual Cure Entities

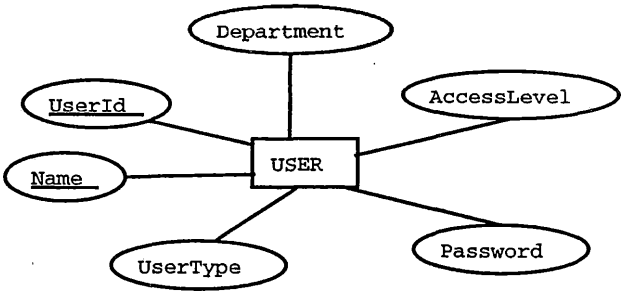


Figure 7a User Entity Properties

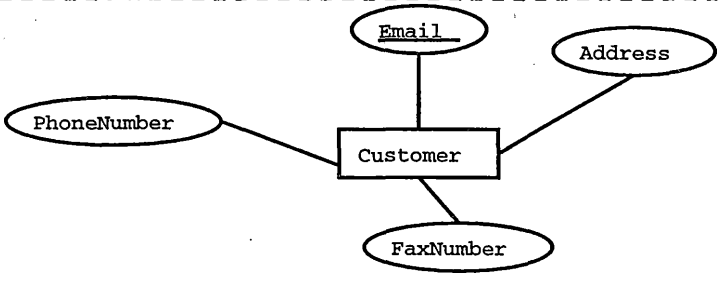


Figure 7b. Customer Entity Properties

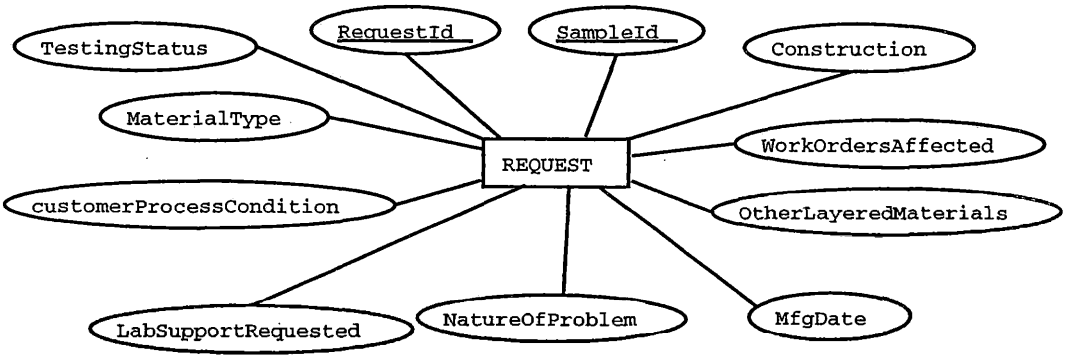


Figure 7c. Request Entity Properties

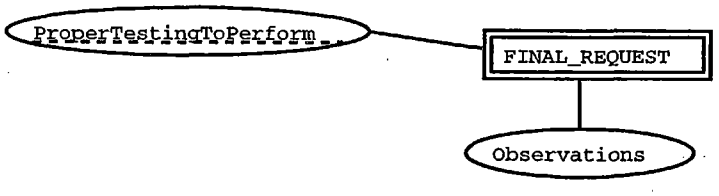


Figure 7d. Final Request Entity Properties

Figure 7. Attributes of: User, Customer, Request, Final Request and Multi-Layer Board Entities

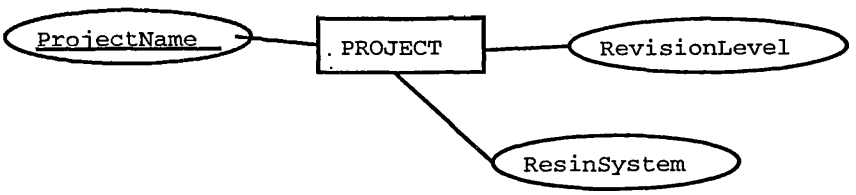


Figure 8a. Project Entity Properties

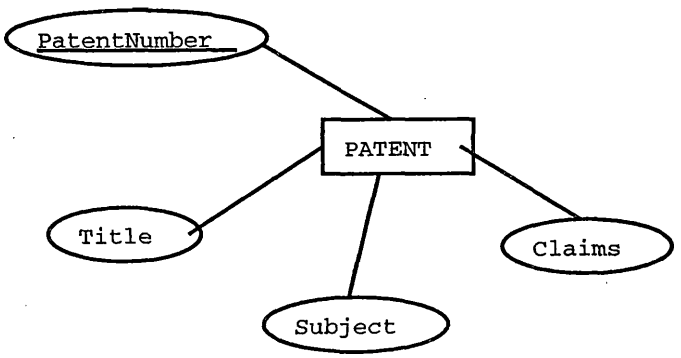


Figure 8b. Patent Entity Properties

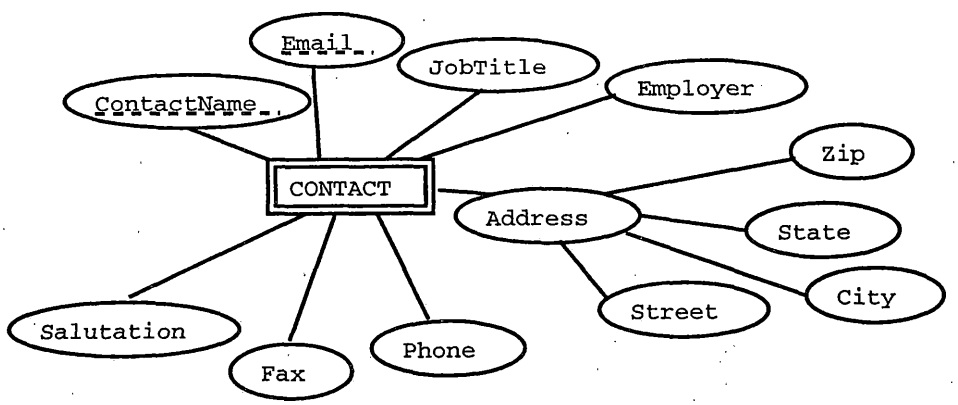


Figure 8c. Contact Entity Properties

Figure 8. Attributes of: Project Patent and Contact Entites

Data Dictionary

This data dictionary defines all classes and attributes of the database management system.

Table 1. Database Dictionary

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
User		A person that uses and has access to theweb-based database management system.	e.g customers, laboratory technicians, technical service representatives and managers.
	user_id	User identification	e.g BenDeveloper MarkTechnician The first word identifies the name of the user e.g Ben, Mark, Speedy. The second one identifies the role e.g Customer, Technician, Developer, Techrep etc. Both words have to start with capital letter. For customers it is just the name of the company: e.g Eurolam, Nelco, etc.
	name	Actual name of the user	e.g Morena Nuno , Rick Marmol. First name and last name
	department	Department in which The user works.	e.g manufacturing quality control sales.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	accessLevel	Privileges granted to the user and this is assigned by the system administrator. There are three categories: 01, 02, 03 and 04.	01: Administrative privileges: Creating accounts, assigning privileges, backups and recovery. 02: Full read and write privileges: The user will read, update, add and remove records from the database: e.g Manager, Technical service Representatives 03: Full read, partial write: The user will read data from the database, and will write to certain tables of the database: e.g Laboratory Technicians. 04: partial read, partial write: the user will read and write to certain tables of the database: e.g Customer
	user_type	Identifies the type of user.	There are four type of users: e.g 'CUSTOMER', 'TECHNICIAN', 'TECHSRVREP' AND 'MANAGER'
	password	Combination of characters and numbers that allows users to access the system. The length of the password will be 7 characters and in lower case.	e.g tyer45y pass342 234errg

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
Customer		A client that buys products from the company (owning the database) which also requests testing on those products.	
	user_id	Name that identifies a customer. Name of a company	e.g Speedy Raytheon CCI
	email	Customer company email address	e.g nelco@aol.com arlon@msn.com speedy@yahoo.com
	address	Customer mailing address: number, street name, city, state, zip code and country.	e.g 16843 FairFax st Fontana CA 92336 USA.
	phone_no	Customer phone number, including country code and country.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)
	fax_no	Customer fax number, including area code.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
Request		Request for a test issued by the customer.	
	request_id	Number that identifies a specific request. It is a number consisting of 6 digits. It is sequential and automatically generated by the DBMS.	e.g 000001 000004 000006
	sample_id	Label that includes characters and numbers which identifies the sample to be tested.	e.g POL0010AAA2 POL: Type of material. POL for polyimide and EPOX for epoxy 0010: Thickness in inches. AAA2:Arbitrary string of characters, length 4, given by the customer
	construction	Amount and type of laminates and prepreg which were used to manufacture the product.	e.g 1 ply of polyimide 1080 prepreg and 1 ply of epoxy laminate.
	workorder_numbers_affected	Series of numbers (consisting of six digits each) and separated by commas. Each number refers to the work order number of the laminates or prepregs that were used to manufacture the multiplayer board.	e.g 1080 polyimide-prepreg: 234334. polyimide laminate: 235464.
	other_layered_materials	Other types of materials included in the laminate or multiplayer board.	e.g Cu, Aluminum etc.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	mfg_date	Date when the laminate or multi-layered board was manufactured.	Must follow this format: 12-JAN-2001
	technician_id	User identification of the technician performing the testing on the request. (Refer to user_id of Technician)	e.g BenTechnician
	test_type	Type of testing to be performed.	e.g tg, t260, t288, t300, electrical strength, dk and df.
	nature_ofthe_problem	Brief summary describing the nature of the manufacturing problem.	e.g Laminate shows delamination, Laminate shows severe blisters.
	lab_support_requested	Type of testing which according to the customer will solve the problem	e.g Tg, T-288 , T-260, T-300, Enthalpy.
	customer_process_condition	Manufacturing conditions done by the customer in a sample. It includes temperature, time and pressure.	e.g 90 minutes @360C at 15 psi kiss pressure.
	material_type	The type of material to be tested. It can be multi-layer board or laminate.	e.g 'MULTILAYER-BOARD', 'LAMINATE'.
	customer_id	User identification of the customer requesting the test. Refer to user_id of Customer	e.g Eurolam, Nelco, etc.
	techsrvrep_id	User identification of the technical service representative working on the request. Refer to user_id of Tech Service Representative.	e.g NidiTechservice, RobertTechservice, etc.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	testing_status	Status of the request	e.g 'SUBMITTED', 'INPROGRESS', 'TESTED' 'COMPLETED'
Tech Service Represen -tative		Technical Service Representative in charge of giving technical support to the customer requesting the testing	e.g Rob Fuller, Peter Kyle
	user_id	Name that identifies a technical service representative. User identification. Refer to user_id of User	e.g NidiTechservice
	email	Technical service representative email address	e.g nelco@aol.com arlon@msn.com speedy@yahoo.com
	address	Technical service representative mailing address: number, street name, city, state, zip code and country.	e.g 16843 FairFax st Fontana CA 92336 USA.
	phone_no	Technical service representative phone number, including country code and country.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	fax_no	Technical service representative fax number, including area code.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)
Final Request		A request issued by the technical service representative based on initial request made by the customer.	
	propertesting_ toperform	The technical service representatives reviews the testing requested by the customer and reviews sample information and technical issue. Based on this information, the technical service representative determines what is the proper testing to perform to solve the issue.	e.g Tg, Enthalpy, T-288, etc.
	observations	Any special conditioning to be performed before testing the sample	e.g Bake samples for 3 hrs @ 430F before testing.
HistoricalData		Records of all completed testing requests.	

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	diagnosis	The technical service representative determines what the problem is, based on test results.	e.g Tg is below 200C. Humidity
	technical_advice	Observations and technical advice given by the technical service representative, based on historical data, test results and experience.	The sample showed some blisters and tg was 210C.therefore the material is considered to be undercured. Advice: Post-bake the material for 1 hr at 430F.
	final_technical_ advice	Technical Advice given by the manager to the customer to solve the manufacturing issue. Technical Advice is based on test results and historical data.	e.g Bake laminates for 2 hrs at 360F.
Testing		Class to generate objects that specify the type of testing to be performed.	
	test_type	Type of testing to be performed	There are 3 types: 'TRANSITION', 'RESISTANCE' and 'RESIDUAL'.
	testingdate	Date in which testing was completed.	e.g Must follow the format : 12-JAN-2001
Transi-tion		Type of testing related to thermal characteristics of the product.	

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	tg	Temperature transition. When materials are exposed to heat, there is a temperature point in which their dimension changes dramatically.That temperature point is called tg.	e.g 200C
	ctex	Dimension change in the x axis when a material is exposed to high temperature.	e.g 5um/mC
	ctey	Dimension change in the y axis when a material is exposed to high temperature.	e.g 7um/mC
	ctez	Dimension change in the z axis when a material is exposed to high temperature	e.g 15 um/mC
	expansion	Percentage that represents the dimension change in a material that has been exposed to high temperature	e.g 10 % This dimension change is measured from 10C to 288C
Resis-tance		Destructive testing that consists of exposing a material to high temperature during certain period of time to check their resistance.	The test stops when the sample to be tested shows delamination or blisters.
	t300	Test in which the material to be tested is exposed for 60 minutes at 300C	e.g PASS/FAIL The test result is represented by a PASS or FAIL note.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	t288	Test in which the material to be tested is exposed for 60 minutes at 288C	e.g Test results are described by: PASS/FAIL notes.
	t260	Test in which the material to be tested is exposed for 60 minutes at 260C	e.g Test results are described by: PASS/FAIL notes
	t300_min	Resistance time to high temperature. The material is exposed at 300C	e.g 25 minutes
	t288_min	Resistance time to high temperature. The material is exposed to 288C	e.g 20 minutes
	t260_min	Resistance time to high temperature. The material is exposed to 260C	e.g 50 minutes.
Residual Cure		Type of testing that measures the amount of energy left in a material after exposure at a given temperature	
	enthalpy	Chemical property that is defined in terms of joules/grams. Energy left in a material after being exposed at 220C.	
MSDS		Manufacturing Safety Datasheet. Required safety datasheet for raw materials and final products that includes safety information for proper handling of the product.	

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	productName	Name of the product	e.g Acetone Toluene
	chemicalName	Name of the product in the scientific community	e.g Methanol, Ethanol
	cas	Certificate of advance study. Specific number given to each MSDS in the scientific community	e.g 984583-345
	purpose	Intent of the document	e.g Avoid health deterioration
	Flashpoint	The lowest temperature in which vapors above a volatile combustible substance ignite in air when exposed to flame.	
	dateReceived	Date in which the material was received in the plant.	e.g 04/15/01
	amount	Amount of material that was received.	e.g 400 pounds
	distribution	The form of distribution	e.g plastic bags, bottles etc.
	evaluation	Indicates if the product is distributed for evaluation purposes.	e.g Evaluation product.
	chemicalFamily	The chemical family to which the product belongs.	e.g Carbonates, etc.
	specificgravity	The ratio of the density of a substance to the density of some substance(as pure as water)taken as a standard when both densities are obtained by weighing in air.	e.g 0.990

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	unit	Units in which the amount will be measured.	e.g lbs, kgs, etc.
Vendor		Name of the company that provides any type of raw material.	
	companyCode	A special code assigned to each vendor. This code is used to refer to a vendor without specifying their complete name.	e.g For Nelco Corporation: Nel3421 Nel: 3 first letters of company name. 3421: Sequential Number.
	supplier	The manufacturing company	e.g Bairco Corporation Technitron.
	division	A group of organisms that form part of a larger group.	e.g Arlon Electronics Division.
	vendorPhoneNumber	Phone number of the vendor, including area code, for international numbers include country code	e.g USA : (909)786-9087 International : (000000000)-(000)-998-9089 (CountryCode)-(AreaCode)-phonenumber.
	vendorFaxNumber	Fax Number of the vendor, including area code.	e.g USA : (909)786-9087 International : (000000000)-(000)-998-9089 (CountryCode)-(AreaCode)-phonenumber.
	vendoremail	e-mail of the vendor	e.g speedy@hotmail.com
	vendoraddress	Address of the vendor, including: number, street name, city, state, zip code and country.	e.g 1243 Arrow Blvd San Bernardino CA 92330 USA.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
Raw Material Request		A request for testing raw materials; raw materials are any products coming from vendors.	
Raw Material		Crude or processed material that can be converted by manufacturing, processing or combining into a new and useful product. Raw materials are bought from vendors	
Engineer		A person who deals with manufacturing issues.	
Mfg Product Request		Any testing request issued on products manufactured within the plant.	
Mfg Prepreg		Any prepreg manufactured within the plant.	
	PartNumber	Conbination of numbers and letters which identifies a specific roll of manufactured prepreg.	e.g POL231360 POL: Type fo resin system. 2313: Type of fiberglass 60: Percentage of resin content.
	LotNumber	Sequential number which identifies a specific roll of prepreg manufactured within the plant.	e.g 234565A01 234565: Sequential Number. A: Machine used. 01: roll number.
Mfg Laminate		Any laminate manufactured within the plant.	
	WorkOrder	Sequential number which identifies a specific laminate manufactured within the plant.	e.g 234454

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	assemblyNumber	Combination of letters and numbers that identify a specific laminate.	e.g POL008AAA07 POL:Type of resin system. 008: Laminate thickness. AAA07: Construction.
	CureTime	The time that the laminate remained in the oven to be cured.	e.g 90 minutes.
	CureTemp	The temperature used to cure the laminate.	e.g 430F
Manager		A person who directs the research and development team, directing projects and authorizing testing to be performed within the lab.	
	user_id	Name that identifies the. User identification. Refer to user_id of User	e.g PaulManager
	email	Manager email address	e.g paulkyle@aol.com
	address	Manager mailing address: number, street name, city, state, zip code and country.	e.g 16843 FairFax st Fontana CA 92336 USA.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	phone_no	Manager phone number, including country code and country.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)
	fax_no	Manager fax number, including area code.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)
Developer		A person who develops products in the polymers industry, working through projects to research and test new chemical components.	
Laboratory Technician		A person who performs all analytical testing (tg, t288, t300, enthalpy, etc) within the laboratory.	

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
Project		A definitely formulated piece of research. Each project generates samples to be tested and tests results are the parameters that indicate where to go in each project.	
	projectName	Name of the project	e.g. High Thermo-conductivity
	resinSystem	Type of resin system in study for example: Epoxy.	e.g Polyimide Epoxy Poly-Epoxy
	revisionLevel	The phase number in which the project is situated	e.g 001
Patent		Of, relating to, or concerned with the granting of patents, specially for inventions, protected by a trademark or a trade name so as to establish proprietary rights.	
	patentNumber	Specific and unique number that identifies each patent in the laboratory	e.g 200100045 2001: year 00045: sequential number
	title	Title of the patent.	e.g Low dielectric constant.
	Subject	Subject of the patent. Brief description explaining the subject of the patent.	e.g A Poly-Epoxy combined with some fillers with the objective of lowering the dielectric constant.
	Claims	Brief description of the claims	e.g It is a new product because It gets readings of 0.0003 df

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
Research And Develop-ment Request		Testing request issued on research and development samples.	
	SampleId	Label that identifies each specific research sample.	e.g 25FR-0001 25FR:Project Name code. 0001: Sample Number.
	LaboratoryNote BookReference	Label that references the laboratory Notebook where further information of the sample to be tested can be found.	e.g 2001BRII-40-01 2001:Year BS:Lab Technician Initials. II:NoteBook number. 40: page number. 01: sample number.
	Varnish	A liquid preparation that when spread and allowed to dry on a surface forms a hard lustrous typically transparent coating. Varnish manufactured within the lab for research purposes.	
Research Prepreg		Prepreg manufactured within the lab for research purposes.	
	construction	Type and amount of prepreg used to manufacture the research and development laminate.	e.g 3 ply Polyimide prepreg. 2 ply Epoxy prepreg.
	cureTimeRL	Time the the research laminated was exposed to high temperature for curing purposes.	e.g 90 minutes.

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	cureTemperatureRL	Temperature at which the research laminate was exposed for curing purposes	e.g 360 F
Contact		A person serving as a source of special information. Business contacts.	
	contactName	Name of the person who is in charge of giving samples, information for ongoing projects.	e.g Rob Fuller
	contactEmail	Email of the contact.	e.g paulkyle@yahoo.com
	jobTitle	The job title of the contact(person who is giving information)	e.g Manager. Laboratory Technician. Sales Representative.
	employer	Name of the company employing the contact	e.g Nelco. Technitron
	contactAddress	Contact address including: number, street name, city, state, zip code and country.	e.g 16849 Foothill Blvd Upland CA 92338 USA
	contactPhoneNumber	Contact phone number including area code, for international numbers include country code.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)-(000)-(0000000) (Country code) (Area code) (phone number)

	Data Item	Definition/Usage	Sample Data Value/Comments
Class	Attribute		
	contactFax	Fax number of the contact, including area code, for international numbers include country code.	For calls within USA. Leave the country code blank. e.g USA (909)854-1503 International (000000000)- (000)- (0000000) (Country code) (Area code) (phone number)
	salutation	The word or phrase for greeting.	As gentlemen or Dear Sir or Madam, conventionally comes immediately before the body of a letter.

Conceptual Design

The Enhanced Entity Relational (EER) model is used to represent the conceptual schema. The EER diagram shows the big picture of the system and the gray area represents the part of the project that is implemented and it corresponds to the tech service area (see Figure 3). This area covers all features that this project proposes.

EER diagram is only used in this project to get a general understanding of the system but the object-oriented model will be used for design, analysis and implementation. Object orientation is a strategy for organizing systems as collections of interacting objects that combine data and behavior. It applies to many

technology areas, including databases. Databases and programs can be developed together for ease of conceptualization, implementation, maintenance, and potential reuse. Thus the object model not only provides a basis for analyzing requirements but provides the genesis for design and implementation. There are several advantages in using object-oriented models for instance:

- Reduced life cycle cost: The clear documentation facilitates maintenance and enables more software reuse to occur-from code libraries, from related projects, and from within a project. [2][6]
- Faster time to market: You can organize large projects into work units that can be assigned to different development teams. This is a large project and the work unit of technical service for customer is the one that is implemented.
- Communication: Object models bring important names and application concepts to the fore so they can be defined and understood by all parties. Models promote communication between developers and customers by separating deep conceptual issues from distracting implementation details. [2]

- Extensibility: Software organized about an object-oriented theme parallels the real world and is flexible with respect to changes in requirements; to a large extent the software may be extended for new requirements without disrupting solutions to existing requirements. In contrast software that is decomposed into arbitrary functions (the procedural approach) is brittle and often difficult to evolve.
- Object-oriented modeling is especially helpful for database applications. An object-oriented database can be regarded as a persistent store of objects created by an object-oriented programming language. With an ordinary programming language, objects cease to exist at program termination. With an object-oriented database, objects persist beyond the confines of program execution. An object-oriented DBMS manages the data, programming code, and associated structures that constitute an object-oriented database.

In contrast to relational DBMSs, object-oriented DBMSs vary widely in their syntax and capabilities. The object-oriented model represents the conversion of the EER

diagram (see Figures 9, 10 & 11). A second object-oriented model is presented for the Technical Service Area that is implemented in this project. The object-oriented model for technical service support provides a uniform abstraction for the design of both programming code and database code. The object-oriented model maps naturally to all major languages and the standard types of DBMSs. Unlike a procedural approach, the same paradigm can be applied throughout analysis, design, and implementation. [1][6]

Logical Design

During this phase we map (or transform) the conceptual schema from the high-level data model (EER diagram and OMT diagram) into the Oracle8 object model (see Figure 9 & 10).

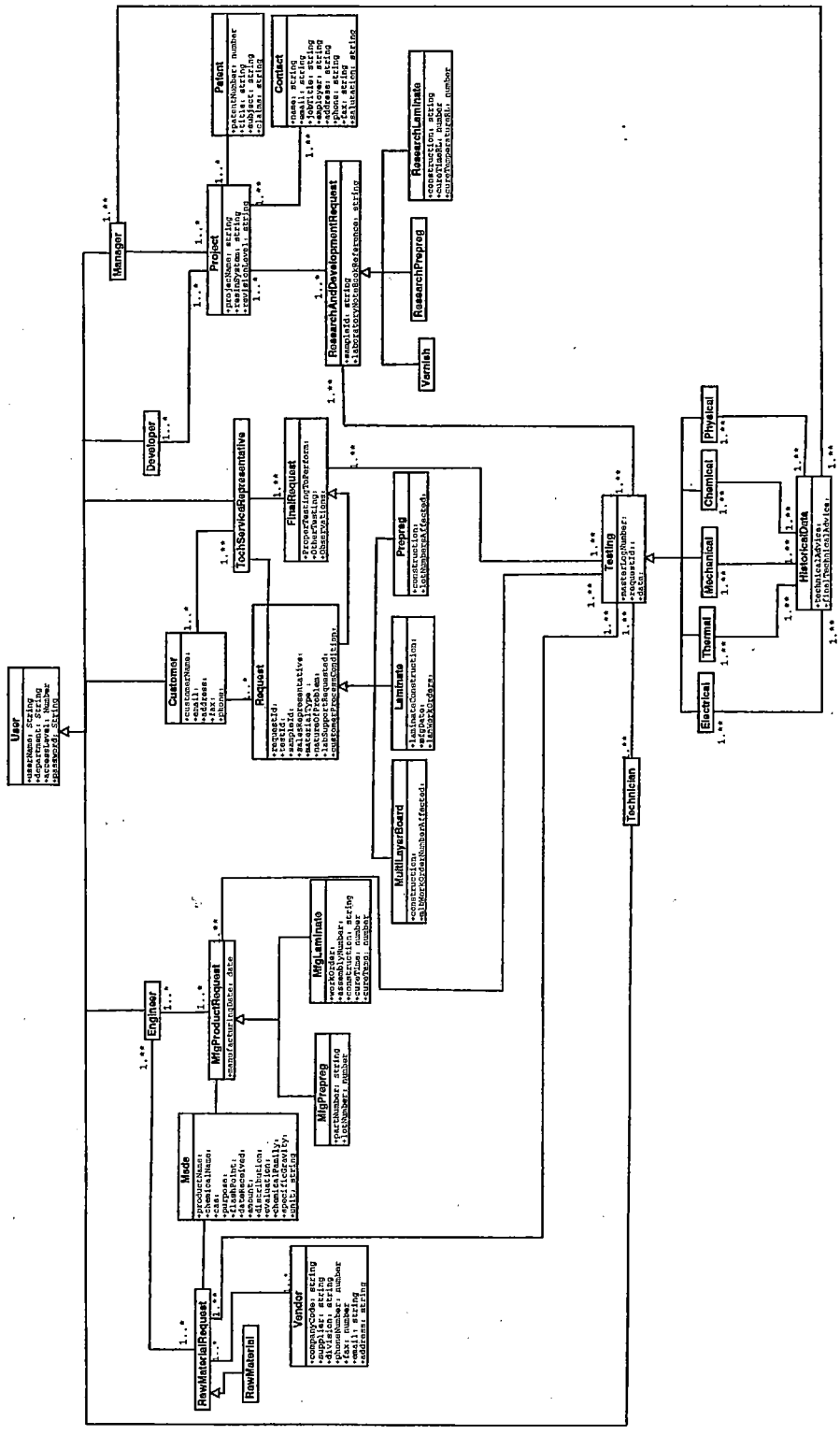


Figure 9. Object-Oriented Model: General Overview

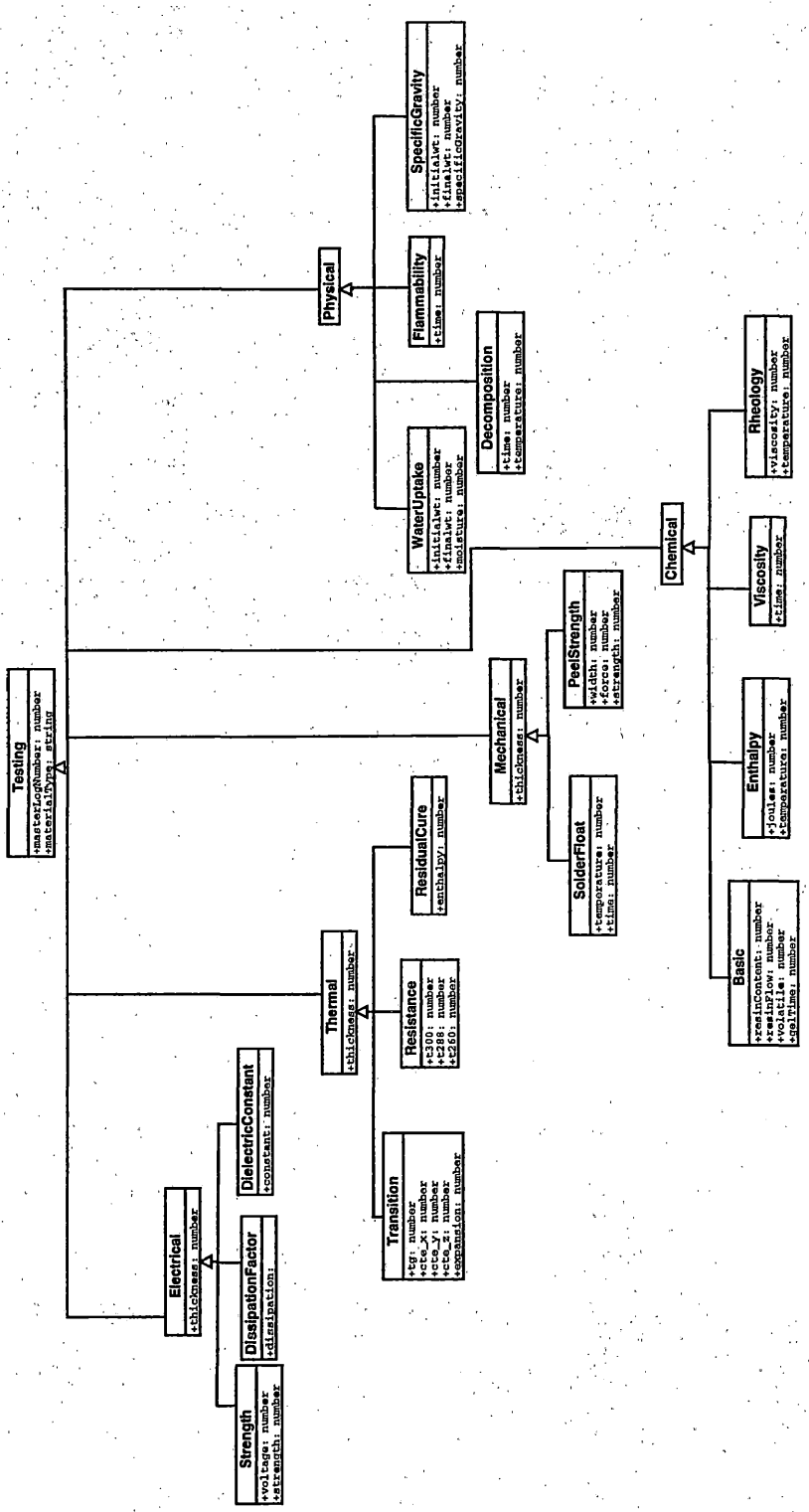


Figure 10. Object-Oriented Model: Technical Service Support

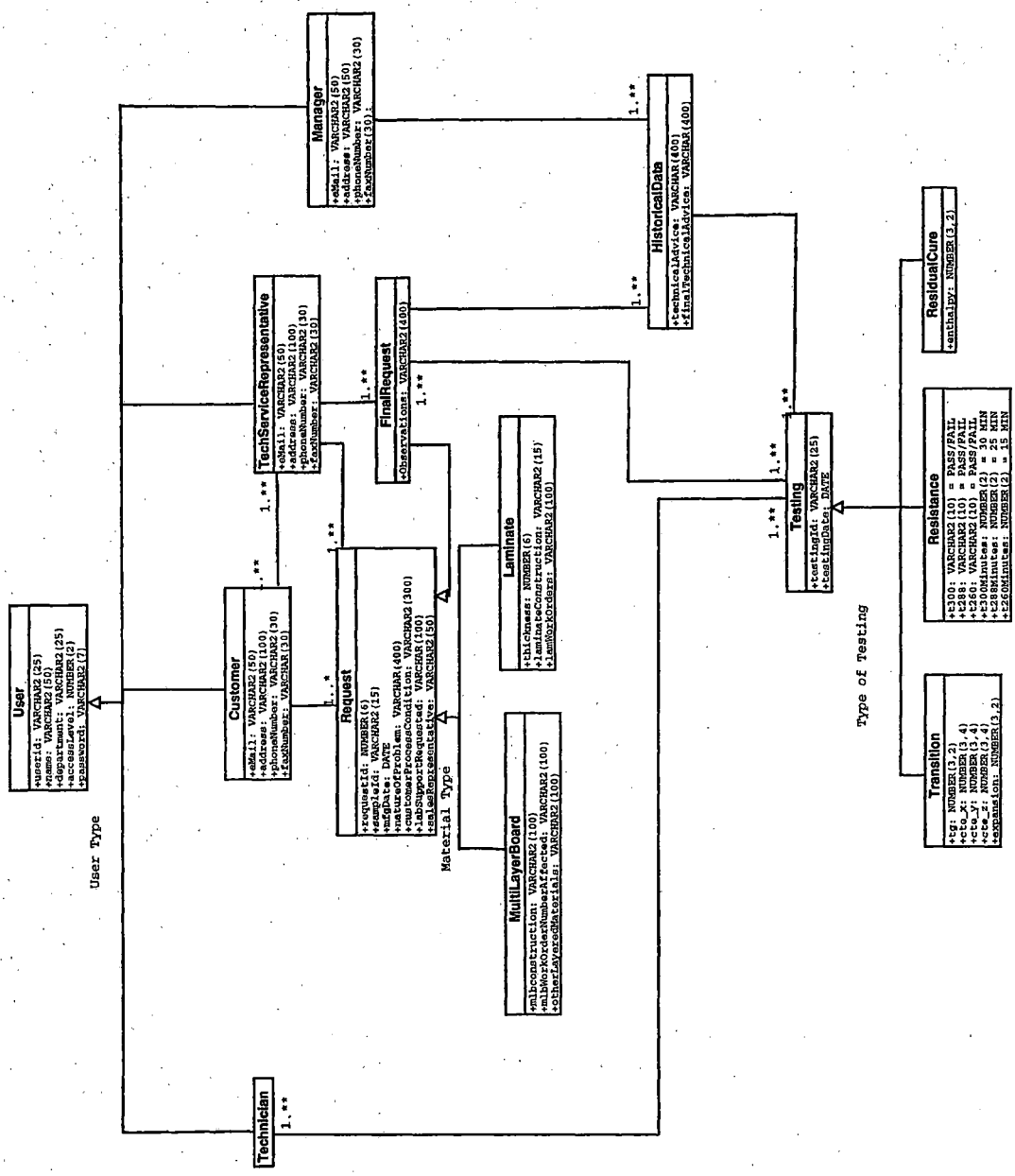


Figure 11. Object-Oriented Model: Testing Unit

Table 2. Relational Database Tables

a. Table name: USER

Column Name	Key Type	Null/ Unique	Check	Data Type	Length
USER_ID	PK, CK	NN		VARCHAR2	25
NAME	CK	NN		VARCHAR2	50
DEPARTMENT		NN		VARCHAR2	25
ACCESS_ LEVEL		NN	01,02, 03,04	NUMBER	2
USER_ TYPE		NN	TECHNICIAN CUSTOMER TECHSERREP MANAGER	VARCHAR2	25
PASSWORD	CK	NN	VARCHAR2	VARCHAR2	7

b. Table name: TECHNICIAN

Column Name	Key Type	Null/ Unique	FK Ref Table	Fk Ref Column	Data Type	Length
USER_ID	PK, FK, CK	NN	USER	USER_ID	VARCHAR2	25

c. Table name: CUSTOMER

Column Name	Key Type	Null/ Unique	FK Ref Table	FK Ref Col	Data Type	Length
USER_ID	PK, FK, CK	NN	USER	USER_ID	VARCHAR2	25
E_MAIL	CK	NN			VARCHAR2	50
ADDRESS		NN			VARCHAR2	100
PHONE_ NUMBER		NN			VARCHAR2	30
FAX_ NUMBER		NN			VARCHAR2	30
TECHSRVREP_ID	FK	NN	TECHSERREP	USER_ID	NUMBER	6

d. Table name: TECHSERREP

Column Name	Key Type	Null/ Unique	FK Ref Table	Fk Ref Column	Data Type	Length
USER_ID	PK, FK, CK	NN	USER	USER_ID	VARCHAR2	25
E_MAIL	CK	NN			VARCHAR2	50
ADDRESS		NN			VARCHAR2	100
PHONE_ NUMBER		NN			VARCHAR2	30
FAX_ NUMBER		NN			VARCHAR2	30

e. Table name: MANAGER

Column Name	Key Type	Null/ Unique	FK Ref Table	Fk Ref Column	Data Type	Length
USER_ID	PK, FK, CK	NN	USER	USER_ID	VARCHAR2	25
E_MAIL	CK	NN			VARCHAR2	50
ADDRESS		NN			VARCHAR2	100
PHONE_ NUMBER		NN			VARCHAR2	30
FAX_ NUMBER		NN			VARCHAR2	30

f. Table name: REQUEST

Column Name	Key Type	Null/ Unique	FK Ref Table	FK Ref Col	Default Value	Data Type	Length
REQUEST_ID	PK,CK1	NN				NUMBER	6
SAMPLE_ID	CK2	NN				VARCHAR2	15
CONSTRUCTION		NN				VARCHAR2	100
WORKORDER_NUMBERS_AFFECTED		NN				VARCHAR2	100
OTHER_LAYERED_MATERIALS		NN				VARCHAR2	100
MFG_DATE		NN				DATE	
TECHNICIAN_ID			TECHNICIAN	USER_ID		VARCHAR2	25
NATURE_OFTHE_PROBLEM		NN				VARCHAR2	400
LAB_SUPPORT_REQUESTED		NN				VARCHAR2	100
CUSTOMER_PROCESS_CONDITION		NN				VARCHAR2	300
MATERIAL_TYPE		NN			MLB LAMINATE	VARCHAR2	16
CUSTOMER_ID	FK	NN	CUSTOMER	USER_ID		VARCHAR2	25
TECHSRVREP_ID	FK	NN	TECHSERREP	USER_ID		VARCHAR2	25
TESTING_STATUS		NN				VARCHAR2	10

g. Table name: FINAL_REQUEST

Column Name	Key Type	Null/ Unique	FK Ref Table	FK Ref Col	Data Type	Length
REQUEST_ ID	PK, FK, CK	NN	REQUEST	REQUEST_ ID	NUMBER	6
PROPER TESTING_ TO PERFORM		NN			VARCHAR2	100
OBSERVATIONS		NN			VARCHAR2	400
USER_ID	FK	NN	TECHSERREP	TECHSERREP_ ID	VARCHAR2	25

h. Table name: HISTORICAL

Column Name	Key Type	Null/ Unique	FK Ref Table	FK Ref Col	Data Type	Length
HISTORICAL_ ID	PK, CK	NN	FINAL_REQUEST	REQUEST_ ID	NUMBER	6
DIAGNOSIS		NN			VARCHAR2	400
TECHNICAL_ ADVICE		NN			VARCHAR2	400
FINAL_ TECHNICAL_ ADVICE		NN			VARCHAR2	400
MANAGER_ID	FK	NN	MANAGER	MANAGER_ ID	VARCHAR2	25

i. Table name: TESTING

Column Name	Key Type	Null/ Unique	Default Value	FK Ref Table	FK Ref Col	Data Type	Length
TESTING_ ID	PK, CK	NN				NUMBER	8
HISTORICAL_ ID	PK, CK, FK	NN		HISTORICAL	HISTORICAL_ ID	NUMBER	6
TESTINGDATE		NN				DATE	
TEST_TYPE		NN	TRANSITION RESIDUAL RESISTANCE			VARCHAR2	10
TECHNICIAN_ID	FK	NN		TECHNICIAN	USER_ID	VARCHAR2	25

j. Table name: TRANSITION

Column Name	Key Type	FK Ref Table	FK Ref Col	Data Type	Length
TESTING_ ID	PK,FK	TESTING	TESTING_ ID	NUMBER	8
TG				NUMBER	(6,2)
CTE_X				NUMBER	(6,3)
CTE_Y				NUMBER	(6,3)
CTE_Z				NUMBER	(6,3)
EXPANSION				NUMBER	(5,2)

k. Table name: RESISTANCE

Column Name	Key Type	FK Ref Table	FK Ref Col	Data Type	Length
TESTING_ ID	PK,FK	TESTING	TESTING_ ID	NUMBER	8
T300				VARCHAR2	4
T288				VARCHAR2	4
T260				VARCHAR2	4
T300_MIN				NUMBER	2
T288_MIN				NUMBER	2
T260_MIN				NUMBER	2

l. Table name: RESIDUAL_CURE

Column Name	Key Type	Null/ Unique	FK Ref Table	FK Ref Col	Data Type	Length
TESTING_ ID	PK,FK	NN	TESTING	TESTING_ ID	NUMBER	8
ENTHALPY					NUMBER	(6,2)

CHAPTER THREE

BROWSER-BASED INTERFACE

Java Server Pages

JSP is a recent language specification developed by Sun Microsystems (cooperatively with other software companies including Oracle) to allow the generation of dynamic content in the HTML pages of a web application. For example, a JSP may perform a query against the database and report the results as an HTML table. The JSP translator converts JSP files into servlets, which can be executed on a web-server that supports a servlet runner. Once deployed, a JSP can be invoked from a browser via an http URL to return the dynamically generated page. Access to object-relational data in SQL tables is provided through JDBC and SQLJ, which are standard frameworks for database connectivity in Java.

Web Application Architecture

The central entity in web applications is the web server. The basic idea of a web server is quite simple - it understands the HTTP request-response protocol and returns pages that are requested by the user through an HTTP URL. Around this simple concept have grown powerful Java-based technologies such as servlets and JSPs.

Servlets and JSP are executed by the web server upon invocation through a URL. The generated results are returned to the browser as part of the response to the HTTP request.

Java Server Pages Programming

The JSP programming model allows web content to be generated dynamically during program execution through Java scriptlets, declarations, and expressions interleaved with the static content in an HTML page. A scriptlet is a block of Java code that is run as part of the servlet that is generated from JSP translation. Compared to servlets, JSPs provide a convenient shorthand, making them simpler and easier to author. Their main intent is to support clean separation of content generation and presentation logic.

Using Java Beans

JSP supports the use of the <jsp:usebean> tag to invoke a modular program entity known as a Java Bean. Java Beans function as reusable elements of component programming. A Java Bean is a Java program that conforms to certain design rules with well-defined semantics that permit dynamic discovery and manipulation of the bean. For example a bean can have properties with accessor methods.

By default, accessor methods can be associated with their respective properties simply by their naming convention - accessors for a property int x are named getX() and setX (int new X). Such implicit rules as well as the facility for explicitly providing information about a bean through aBeanInfo class support well-defined behavior of the bean. This information can be effectively used by environments in which the bean is embedded. Using Java Beans in a JSP allows clear separation of Java logic that generates dynamic content and its Presentation. Java Beans were used in most of JSP files created for this project.

Database Access Schemes in Java

Two different frameworks, JDBC and SQLJ, are available for connecting to database in a Java program.

In this project JDBC was the main database scheme used for implementation purposes, SQLJ was used in a module for comparison purposes.

Java Database Connectivity

Java programs can interface with any database that can be accessed using Structured Query Language, using classes that come with the JDK. The Java Database Connectivity (JDBC) class library provides a standard way

for establishing and maintaining a Java program's connection to a database. Once a connection to the database is established, SQL can be used to access and process the contents. [4][12]

The JDBC library was designed as an interface for executing SQL statements, and not as a high-level abstraction layer for database access. So, although it wasn't designed to automatically map Java classes to rows in a database, it allows large scale applications to be written to the JDBC interface without worrying too much about which database will be deployed with the application.

The JDBC architecture is based on a collection of Java interfaces and classes that together enable you to connect to data sources, to create and execute SQL statements, and to retrieve and modify data in a database. These operations are illustrated in the figure below.

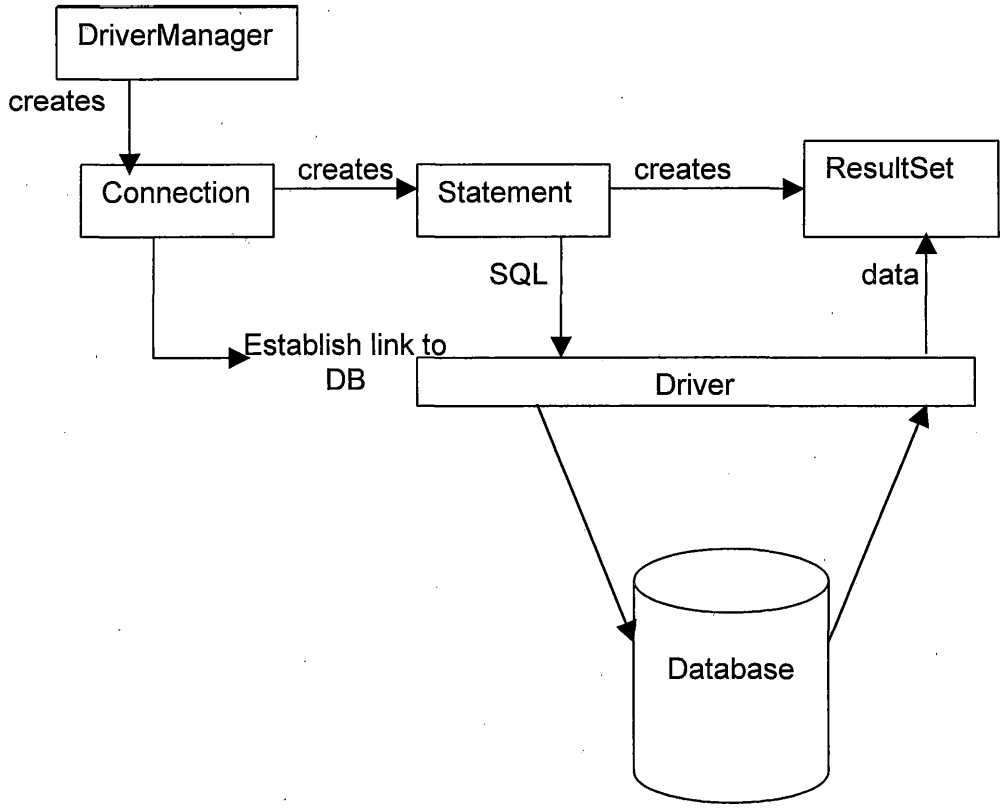


Figure 12. Java Database Connectivity Architecture

Each of the boxes in the illustration represents a JDBC class or interface that has a fundamental role in accessing a relational database.

Oracle Access with Java Database Connectivity

Java is designed to be platform independent. A pure Java program written for a Linux machine will run without recompilation on a Windows machine, an Applet Macintosh, or any platform with the appropriate Java virtual machine. JDBC extends this to databases. A program written using

JDBC will also run in any platform. JDBC is written using Java code, and leaves the platform (database) specific code to the driver.

When the database is changed, the driver needs to be changed and the code will be ready to run. The Java JDBC code is portable because the database specific code is contained in a Java class known as the driver. The two most common types of drivers are: the thin driver and the OCI driver. The thin driver is also known as a Type IV driver; it is a pure Java driver that connects to a database using the database's native protocol. Thin drivers can be used in any environment, and is intended for use in Java applets and other client-sideprograms. A java client can be run on any platform. For this project a thin driver is used since the access to the database will be through Java Server Pages, the JDBC driver downloaded with an applet or used by a Java client may not have access to platform native code and must be pure Java. [12]

The Oracle JDBC 1.0 drivers are in a Java archive file name classes111.zip and it is normally located in the $ORACLE_HOME/jdbc/lib directory.

The Java requirements are in the java.sql package. This package comes standard with the JDK 1.1.

Java Database Connectivity Basics

All Java programs that access a database follow the three main following steps:

- Load the driver and make a connection to the database
- Execute a SQL statement against tables
- Access and use the results of the SQL statement

Structured Query Language in Java Code

SQLJ is a recent ANSI standard for embedding static SQL statements directly in Java code [20]. The mixed code is converted to Java by the SQLJ translator, and can be executed on a database using the SQLJ runtime library and an underlying JDBC driver. Oracle is a major participant in the design and development of SQLJ, and supports SQLJ wherever JDBC runs.

SQL statements in SQL are static, that is, they must be known at program time and cannot change as the program executes. Data values passed to SQL operations can be determined at runtime but the SQL operation is known a priori. In contrast, the JDBC API is fully dynamic-the SQL statement itself can be formulated on the fly. Most SQL operations in a typical database application are static. SQLJ provides a simpler model for static SQL statements

compared to JDBC, and provides a higher-level interface by automatically managing JDBC statement handles. Additionally, the SQLJ translator can check the SQL statement against a database for syntax and semantic errors. This checking is performed at compile time unlike just at runtime as in JDBC, and it is independent of the actual flow of program logic.

Java Server Pages Design and Implementation

After creating all relational database tables a JSP file was created for each feature in the users (customer, technical service representative, laboratory technician, manager) menus (see Appendices D-G).

CHAPTER FOUR

WEB-SITE DESIGN

Requirements Collection and Analysis

The object model was extracted from the main object model (see Figure 9). The web-site design will be specific for the technical support area (see Figure 10).

Use Case Diagram

Use cases are used to capture typical scenarios to help understand the requirements. [6] A use case is a typical interaction between a user and a computer system. Figure 13 shows the interaction among different actors in the web site and the functions that they can perform using the system.

Technical Support System Design

The main reason for creating the web site is to allow users to have access to the database management system, from anywhere using the Internet. The browsers used are:

Netscape 4.7. and Microsoft Internet Explorer 5.5.

The main concern of this web-site is functionality so that information can be accessed and shared in an efficient and timely manner.

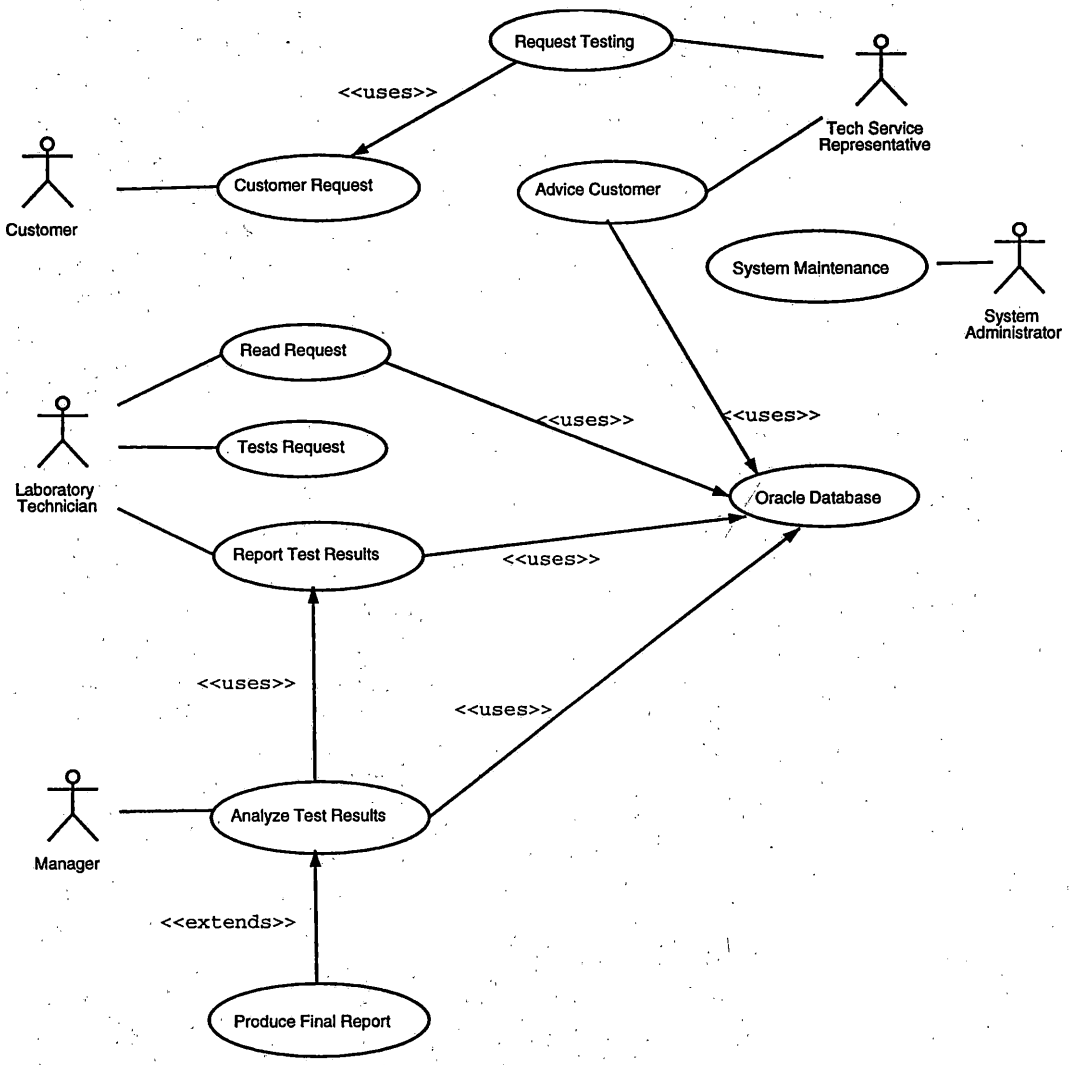


Figure 13. Web-Site Use Case Digram

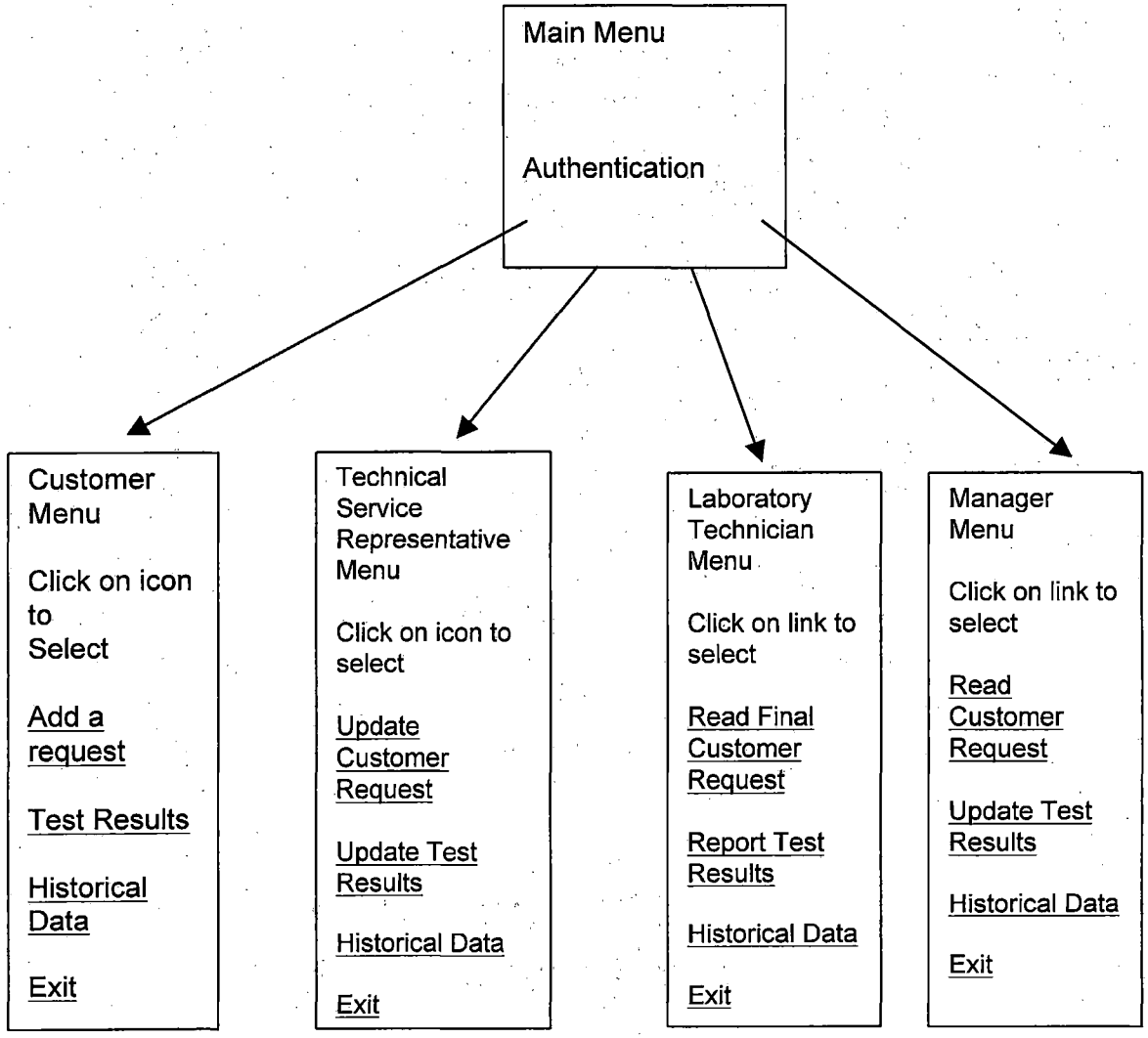


Figure 14. Technical Service Support System

CHAPTER FIVE

IMPLEMENTATION

Database Implementation

The object-oriented model provides a useful approach to design relational database applications, because object oriented models are expressive, concise, and easy to develop. A relational database is a database in which the data is logically perceived as tables.

A relational database is self-descriptive, since the data dictionary contains data that describe relational tables.

The database management system used to implement the technical service support system was Oracle 8i. All tables from the conceptual design were created with its respective constraints, including primary, unique and foreign keys (see Appendix A).

Java Server Pages

Java Server Pages were done using scriptlets (Java code) embedded in HTML formatting code. Java Beans were also used for implementation purposes (see Appendix F). The database access schemas used were JDBC and SQLJ, all embedded in JSP files.

The Technical Service Support System (Web-Site), which is accessed using a browser as a front end was developed using JSP files and it was tested using Netscape 4.7. and Explorer 5.5.

Web-Site

Apache Server

A pre-built package of the apache server was installed on Linux 6.2 Operating system. First the binary package is unpacked.

```
# cd /tmp
# zcat apache_1.3.12-i386-whatever-linux2.tar.gz | tar xf -
# cd apache_1.3.12
```

The location of the Server Root was set up using the following command:

```
# ./install-bindist.sh /usr/local/web/apache
```

Then the binary and documentation was installed as follows:

Apache Server Set Up

Table 3. Apache Server Set Up

Directory Tree	Location
Apache Source tree	Not installed
Apache Server Root	/usr/local/web/apache
Apache Document Root	/usr/local/web/apache/htdocs

Then the http.conf file is edited to make sure configuration is set up properly, mainly for the ip address and security issues.

To start the server the following command has to be executed.

```
$ cd /usr/local/web/apache/bin
$ apachectl start
httpd started
```

Then the JSP files for the technical service support system (Web-Site) were placed in the Apache Document root.

Web-Site Key Path

See Appendices D-G

CHAPTER SIX

ANALYSIS

Java Database Connectivity versus Structured Query Language in Java Code

Two different frameworks, JDBC and SQLJ, are available for connecting to database in a Java program.

Java Database Connectivity

The JDBC specification from Sun Microsoft defines a set of interfaces for SQL-based database access from Java. JDBC is a call-level application interface (API), which means that SQL statements are executed on a database by calling methods in the JDBC library from the Java program. Database vendors can provide different implementations of the JDBC APIs for their databases. For example, Oracle provides three JDBC drivers [3], namely the JDBC-OCI driver which uses the client-side OCI installation to interact with the Oracle database, the JDBC thin driver that is written purely in Java and communicates directly with the database using Java sockets over TCP/IP, and a JDBC server-side driver packaged as part of the Oracle 8i Jserver for executing Java store procedures inside the database. Which JDBC driver is appropriate for the application depends on the deployment requirements. The

important point is that all JDBC drivers support the same standard set of interfaces, thereby promoting portability across different JDBC drivers as well as databases.

The JDBC programming model is based on ODBC. Interfaces defined in the java.sql package represent database connections (java.sql.Connection), statement execution handles (java.sql.Statement), and query result sets (java.sql.ResultSet), among others. Resources provide row-by-row access to the results returned by a SQL query.

Structured Query Language in Java Code

SQLJ is a recent ANSI standard for embedding static SQL statements directly in Java code [5]. The mixed code is converted to Java by the SQLJ translator, and can be executed on a database using the SQLJ runtime library and an underlying JDBC driver. Oracle is a major participant in the design and development of SQLJ, and supports SQLJ wherever JDBC runs.

SQL statements in SQLJ are static, that is, they must be known at program time and cannot change as the program executes. Data values passed to SQL operations (i.e. the values of bind parameters) can be determined at runtime but the SQL operation is known a priori. In contrast, the JDBC API is fully dynamic that allows the SQL statement

itself to be formulated "on the fly". Most SQL operations in a typical database application are static. SQLJ provides a simpler model for static SQL statements compared to JDBC, and provides a higher-level interface by automatically managing JDBC statement handles.

Additionally, the SQLJ translator can check the SQL statements against database for syntax and semantic errors. This checking is performed at compile-time in contrast to runtime as in JDBC, and it is independent of the actual flow of program logic. Compared to JDBC, SQLJ programs are therefore more robust, much quicker to write and easier to maintain.

Summary of Differences between Java Database Connectivity and Structured Query Language in Java Code

The following section presents source code of a Java bean implemented using both SQLJ and JDBC. This Java Bean retrieves all the information of a testing request, including testing status. This bean is used in the Manager submenu where the manager has the option to retrieve information of a specific request to know the status of the request.

JDBC	SQLJ
Imports and Declarations Import the sql library for java and declare JDBC classes import java.sql.*; Connection conn = null; Statement stmt = null; ResultSet rset = null;	Imports and Declarations Import the sql library for java and SQLJ libraries. Declare the dc (Default Context) import java.sql.*; import sqlj.runtime.ref.DefaultContext; import oracle.sqlj.runtime.Oracle; DefaultContext dc = null;
Connection String connStr = "jdbc:oracle:thin:@localhost:1521:ora1"; Connection conn = null; try { if (conn == null) { DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver()); conn = DriverManager.getConnection(connStr,"ben28","benito29"); }	Connection if(dc == null) dc = Oracle.getConnection("jdbc:oracle:thin:@localhost:1521:ora1","ben28","benito29");
Handling SQL statements SQL statements are executed on a database by calling methods in the JDBC library from the Java Program stmt = conn.createStatement(); String query = "SELECT sample_id as \"SampeId\", " +"construction as \"Construction\"," +"workorder_numbers_affected as \"WorkOrderNumber\"," +"other_layered_materials as \"OtherLayeredMat\"," +"mfg_date as \"ManufacturingDate\"," +"technician_name as \"TechnicianName\"," +"lab_support_requested as \"LabSupportRequested\"," +"nature_of the_problem as \"NatureOfTheProblem\"," +"customer_process_condition as \"CustomerProcess\"," +"material_type as \"Material Type as \"MaterialType\","	Handling SQL statement SQL statements are directly embedded in Java code #sql [dc] { SELECT sample_id, construction, workorder_numbers_affected, other_layered_materials, mfg_date, technician.name, lab_support_requested, nature_of the_problem, customer_process_condition, material_type, customer.name, techsrvrep.name, testing_status, date_requested INTO :sample_id, :construction, :wor_num_aff, :other_layered_materials, : mfg_date, : technician_id, : nature_of the_problem : lab_support_requested, :customer_process_condition,

JDBC	SQLJ
+'customer_name as \"CustomerName as \"CustomerName\"," +"techsrvrep_name as \"TechServRepName as \"TechSrvRep\"," +"testing_status as \"TestingStatus\"," +"date_requested as \"DateRequested\"," + " FROM request r, userdbms techsrvrep, userdbms " + " customer"+ " userdbms technician" + " WHERE " + "request_id = :requested" +" AND techsrvrep.user_id = r.techsrvrep" + " AND customer.user_id = r.customer_id" +"AND technician.user_id = r.technician_id" +"AND testing_status <> 'COMPLETED' "; + " WHERE r.request_id=fr.request_id AND " + "fr.request_id=h.historical _id AND " + "r.request_id='" +requestid+ "'"; rset = stmt.executeQuery (query);	: material_type, :customer_id, : techsrvrep_id :testing_status :date_requested FROM request r, userdbms techsrvrep, userdbms customer, userdbms technician WHERE request_id = :requestid AND techsrvrep.user_id = r.techsrvrep AND customer.user_id = r.customer_id AND technician.user_id = r.technician_id AND testing_status <> 'COMPLETED' };
Result Set Query Results must always be handled via ResultSets irrespective of whether they contain one or more rows public static String format(ResultSet rs) throws SQLException { StringBuffer sb = new StringBuffer(); if (rs == null || !rs.next()) sb.append("No matching rows.\n"); else { sb.append(" \n"); ResultSetMetaData md = rs.getMetaData(); int numCols =	Result Set It is possible to fetch a single record into Java variables. sb.append("<BLOCKQUOTE><BIG><B><PRE>\n"); sb.append("Request Id: " + requestid + "\n"); sb.append("Sample Id: " + sample_id + "\n"); sb.append("Construction: " + construction + "\n"); sb.append("Work Order Numbers Affected: " + wor_num_aff + "\n"); sb.append("Other Layered Materials: " +

JDBC	SQLJ
md.getColumnCount(); for (int i=1; i<= numCols; i++) { sb.append("" + *md.getColumnLabel(i) + "");* } do { sb.append("n"); for (int i = 1; i <= numCols; i++) { sb.append(""); Object obj = rs.getObject(i); if (obj != null) sb.append(obj.toString()); sb.append(""); } sb.append(""); } while (rs.next()); sb.append(""); } return sb.toString(); } }	other_layered_materials + "\n"); sb.append("Manufacturing Date: " + mfg_date + "\n"); sb.append("Laboratory Technician Working on the request: " + technician_id + "\n"); sb.append("Nature Of the Problem: " + nature_ofthe_problem + "\n"); sb.append("Testing Requested by the Customer: " + lab_support_requested + "\n"); sb.append("Customer Process Conditions: " + customer_process_condition + "\n"); sb.append("Material Type: " + material_type + "\n"); sb.append("Customer Identification: " + customer_id + "\n"); sb.append("Technical Service Representative: " + techsrvrep_id + "\n"); sb.append("Testing Status: " + testing_status + "\n"); sb.append("Date Requested: " + date_requested + "\n"); sb.append("</PRE></B></BIG><BLOCKQU OTE>"); return sb.toString(); } catch (SQLException e) { return ("Invalid Request Id ! SQL Error: " + e.getMessage() + ""); } } }

JDBC	SQLJ
Closing statements try { if (rset!=null) rset.close(); if (stmt!= null) stmt.close(); if (conn!= null) conn.close(); } catch (SQLException ignored) {}	Closing statements Public synchronized void valueUnbound (HttpSessionBindingEvent event) { if (dc != null) { try { dc.close(); } catch (SQLException ignored) {} } }
The JDBC API is fully dynamic- the SQL statement can be formulated "on the fly"	SQL statements in SQLJ are static, that is, they must be known at program time and cannot change as the program executes. Data values passed to SQL operations (i.e. the values of bind parameters) can be determined at runtime but the SQL operation is known a priori.
Complex model for handling static SQL statements.	Simpler model for static SQL statements, provides a higher level interface by automatically managing JDBC statement handles
Checking for syntax and semantic errors of SQL statements is performed at run-time and it is not independent of the actual flow of the program logic	The SQLJ translator can check the SQL statements against a database for syntax and semantic errors. This checking is performed at compile-time and it is independent of the actual flow of program logic.
JDBC programs are less robust, more difficult to write and maintain	SQLJ programs are more robust, much quicker to write and easier to maintain

An important point is that SQLJ and JDBC are complementary approaches. The web-based database management system was developed using static SQL statements, so that SQLJ and JDBC were used interchangeably. SQLJ is designed to inter-operate with

JDBC; for example, a single database connection can be easily shared across the two programming models.

Likewise, JDBC result sets can be converted to SQLJ iterators and vice-versa. In this project SQLJ was used when retrieving single records e.g. when the manager retrieves information of a specific submitted request to check the status or progress of the request and when the customer retrieves test results of a specific testing request.

Performance Analysis

There are two important performance issues about the database access logic in the SQLJQuery JSP: Database connections. Each time it is invoked with a new search condition, a new database connection is opened by the runQuery() method. In practice, a real web environment would have performance optimizations such as database connection pooling in place. These optimizations are usually applicable in a transparent fashion, e.g., the close() method on a connection may simply return the connection to a shared pool. Thus the JSP would still have the same code but underlying layers would cause available connections to be shared effectively among multiple concurrent users of the database.

Query re-parsing: Another apparent source of inefficiency is the re-parsing of the SQL statement if the JSP is invoked multiple times to perform the same SQL query with different parameters. SQLJ runtime will automatically cache and re-use JDBC statement handles as long as the underlying database connection is open. If connection pooling is in place then these statements handles may even be shared across different users. An alternative to this scheme is to use a session-scoped query bean that explicitly re-uses a prepared statement handle for the duration of the HTTP session.

A web application must manage resources acquired during its execution, such as database connections and JDBC statement handles. In the case of JSPs, there are basically two ways to handle web application resources:

(1) Build the management logic into beans called from the JSP. For example, a session-scoped query bean could acquire a database cursor when it is instantiated and release it when the HTTP session is terminated (either explicitly or implicitly via timeout). However, in this scheme a bean needs to be aware that it is running in a servlet environment; for example, the java.servlet.http.HttpSessionEventListener

interface would have to be implemented by the bean so that it can be notified by the servlet execution engine and release the resources upon expiration of the HTTP session. This is the approach used in this project.

(2) Have the JSP code manage the resources itself. JSPs and servlets in the web application know that they will be running in an HTTP environment, hence they can allocate and de-allocate resources at appropriate times. For example, a JSP can have explicit logic to associate a result set with the HTTP session.

For Oracle JSPs there is an extension that lets a JSP easily control resources lifetimes using well-known lifecycle events.

The SQLJ Query and JDBC Query JSPS have a drawback that Java logic is interspersed with the HTML code. While this approach may be adequate for simple database operations, it may not be suitable for complex database operations like in this project. It make sense to cleanly separate the logic for generation of dynamic content from its presentation. Thus the first method (session scoped Java Beans) was used for implementation.

Validation

The web-based database management system validation consists of critical tests where results are evaluated against the original design, specifications and intended functionalities. The main purpose of the web-based database management system is to have assurance about the quality of the system.

The software testing includes checking that all buttons work as expected in the web-site implementation, checking for normal and abnormal termination, verifying the handling of all valid input data types, check data retrieval times and verifying the handling of error conditions.

The following table shows all testing performed on all tables of the Oracle database.

Table 4. Oracle Database Tables Testing

Oracle Database Table	Tests Performed	Results
USER	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
TECHNICIAN	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS

Oracle Database Table	Tests Performed	Results
CUSTOMER	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
TECHSERREP	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
MANAGER	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
REQUEST	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
FINAL_REQUEST	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
HISTORICAL_DATA	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
TESTING	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
TRANSITION	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS

Oracle Database Table	Tests Performed	Results
RESISTANCE	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS
RESIDUAL_CURE	Attributes Data types Constraints Normalization Primary, Foreign and Candidate keys	PASS

Table 5. Java Server Pages Testing

Java Server Pages	Test Performed	Results
1. Add Request (customer)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
2. Test Results (customer)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
3. Historical Data (customer)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
4. Update Customer Request (tech service rep)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
5. Update Test Results (tech service rep)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS

Java Server Pages	Test Performed	Results
6. Historical Data (tech service rep, lab technician, and manager)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
7. Read Final Customer Request (lab technician, manager)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
8. Report Test Results (lab technician)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS
9. Update Test Results (manager)	Test Java Beans. Test HTML code Test scriptlets Test jps files Test Queries Test Queries Results	PASS

Web-site Paths Testing

The following table shows testing performed on all possible paths in the web-site for the technical service support system:

Web-site Paths Testing

Table 6. Web-Site Paths Testing

Menu	Submenu	Tests	Results
Main Page	Customer	Check for: User authentication information. If the user enters an invalid user id or password, a message indicating so will be displayed. Check that the correct sub menu is associated with the user type i.e. the manager menu must be displayed when the user is a manager submenu. The following checks apply for all paths: Check for normal termination: The user uses the submit, reset or home buttons. This check applies for all web-site paths. Check for abnormal termination: The user uses the back and forward browser features to navigate through the system.	PASS
	Laboratory Technician		
	Technical Service Representative.		
	Manager		

Menu	Submenu	Tests	Results
Customer	Add a Request	The user submits the forms either with all or some empty entries. The system displays a red arrow and a red message indicating the missing entries The user enters an incorrect date format	PASS
	Request Test Results	The user submits the request with an empty entry for request id The user submits the request with an invalid request id.	PASS
	Historical	The user submits the request without selecting any given option The user submits the request without selecting some given options The user submits the request with an invalid date format	PASS
	Exit	The user exits successfully	PASS

Menu	Submenu	Tests	Results
Laboratory Technician	Read Customer Request	Verify that the system assigns the testing request with the highest priority (using date as a parameter) in the waiting list. If there are no testing requests to be assigned, a message is displayed indicating so.	PASS
	Report Test Results (and release for tech service representatives review)	Verify that the system lists all the testing requests assigned to the laboratory technician. Check for errors when: The user enters a wrong request id The user submits the test report form with all entries empty The user enters an invalid data type in all or some entries where number data types are required for i.e. Enter a string when a number data type is required for tg.(Oracle sends error message)	PASS
	Historical Data	Same checks as for customer historical submenu Check for errors when the user enter a number when a string data type is required for customer name entry.	PASS
	Exit	The user successfully exits the system, using the exit button.	PASS

Menu	Submenu	Tests	Results
Technical Service Representative	Update and Release Customer Request	Verify that the system displays all testing requests that have been submitted by the customers. Check for errors when: The user enters an invalid request id The user does not enter a request id The user does not select any test to be performed The user does not enter any observations.	PASS
	Update Test Results and Release for manager review	Verify that the system displays all testing request that that have been already tested and reported by the laboratory technicians Verify that the system displays the correct information for the request id selected. Check for error when: The user selects an invalid request id or does not enter a request id.	PASS
	Historical Data	Same checks as for laboratory technician historical data	PASS
	Exit	Verify that the user exits successfully, using the exit button.	PASS

Menu	Submenu	Tests	Results
Manager	Read Customer Request	Verify that the system displays all testing requests that are in process to be completed, showing their status	PASS
	Update and Release Test Results	Verify that the system displays all testing requests which testing have been completed and which have been updated by technical service representatives Check for errors when: The user enters an invalid request id Verify that when the testing request has been updated, it is release for the customer. Verify that an e-mail is sent to the customer when the user selects the send e-mail button.	PASS
	Historical Data	Same checks as for: Historical Data of Laboratory Technician Menu	PASS
	Exit	The user exits the system successfully using the exit button.	PASS

CHAPTER SEVEN

FUTURE WORK AND CONCLUSION

Future Work

- The suggested next phase of this project is the implementation of the other two areas of the web-based database management system: research & development and manufacturing testing. The design part for these two areas (including Object Models) has been done in this project.
- Preparing context-sensitive online help for the application related to this type of industry.
- Graphical interfaces (Java applets) to display historical data in a graph.
- Real Time application to retrieve test results from test instruments and store results in the database.
- Use design methodology and technologies for implementation for different type of applications (e.g. e-commerce).

Conclusion

The web-based database management system was developed with the main goal of providing a system to improve the storage & retrieval of data and the decision

making in research and development laboratories. Information has to be efficiently (1) stored for fast retrieval, and (2) updated to support all type of activities of a research and development laboratory: manufacturing, technical service support and development of new products.

A web-based interactive system was created to help these type of laboratories to reduce administrative time and costs by avoiding duplicate work. The system will help to streamline management process and increase work efficiency due to a faster response to customer complaints and manufacturing issues.

Modern modeling techniques and technologies were used to design and implement the system. EER diagrams and Object Oriented Models were created to have a better understanding of the system. Finally using object-oriented modeling techniques were used to implement in Oracle8i the final object-relational database.

The web-based database management system was implemented with a two-tiered approach to database access. There is a front-end portion of the application that requests data -- the user using the browser to request data. The data is stored on and retrieved from a server. JDBC and SQLJ were used as database access schemas,

concluding that SQLJ and JDBC are complementary approaches. The web-based database management system was developed using static SQL statements, so that SQLJ and JDBC were used interchangeably. SQLJ is designed to interoperate with JDBC; for example a single database connection can be easily shared across the two programming models.

The SQLJ Query and JDBC Query JSPs have a drawback that Java logic is interspersed with the HTML code. While this approach may be adequate for simple database operations, it may not be suitable for complex database operations like in this project, it makes sense to cleanly separate the logic for generation of dynamic content from its presentation, using session scoped Java Beans.

Future Work for this project includes:(1) The implementation of the other two areas (manufacturing and research and development of new products.(2) Provide context-sensitive online help. (3) Design and Implement java applets that display historical data in a graph and (4) Real Time Applications to retrieve test results from test instruments since all testing is computer based.

CHAPTER EIGHT

GLOSSARY

Access Control - Any form of security device or entry system which is designed to restrict hackers or unauthorized users from entering certain sections of a computer system. Control methods may include the use of passwords.

Applet - On the internet, a small, job-specific software program (application) that is designed to execute a single function. The applet would reside at the internet server, and it would be sent to the user upon request or requirement. Once sent to the user, it is executed and then discarded so that it does not clutter the user's computer (or workstation).

DBMS - Database Management System.

Developer - A person who develops products in the polymers industry, working through projects to research and test new chemical components.

Engineer - A person who deals with the design and improvement, and installation of integrated systems (as of people, materials, and energy) in industry. In this specific project a person who deals with manufacturing issues.

Java Database Connectivity (JDBC) - The programming interface that defines how Java programs can use the structured query language to query a database.

Laminate - A product made by bonding together two or more layers of material. In this case it could be prepregs or laminates.

Manufacturing Product - Any item produced as if by manufacturing, for instance, laminates, prepregs etc.

MSDS (Manufacturing Safety Datasheet) - Required safety datasheet for raw materials and final products that include safety information for proper handling of the product.

Oracle - A database management system developed by Oracle Systems Corporation to enable users to access data from large corporate databases. The word oracle comes from the Latin word orare, meaning speak.

Project - A definitely formulated piece of research. Each project generates samples to be tested and the tests results are the parameters that indicate where to go in each project.

Patent - of, relating to, or concerned with the granting of patents specially for inventions. protected by a trademark or a trade name so as to establish

propietaryhts analogous to those conveyed by letters patent or a patent.

Prepreg - Prepreg is a shorthand expression for "pre-impregnated." It is fiberglass or other fabric which we have saturated (impregnated) with a polyimide, epoxy or other resin system, which has been partially "cured" (or reacted) during the coating operation. Prepreg is also called "B-Stage".

Raw Material - Crude or processed material that can be converted by manufacture, processing, or combination into a new and useful product.

Technician - A specialist in the technical details of testing chemical properties of polymers.

Tech Service Laminate - Laminate that is in customer possession and is returned either because of routine testing or because of a customer complaint.

Tech Service Prepreg - Prepreg that is in customer possession and is returned either because of routine testing or because of a customer's complaint.

Tech Service Sample - Sample submitted by tech service representatives to be tested in the laboratory. It can be laminate, prepreg or multiplayer board.

User - A person that utilizes the web based database management system for the purpose of data processing and information exchange.

Vendor - One that vends crude or processed materials.

Web Browser - A Software program that enables a user to access files from any computer that is connected to the internet.

Web Page - On the internet, a unit of data that representsinformation in the form of text and/or graphics. In this context, size is not defined, in so far as a page could be single word or a volume of books residing at an internetdomain.

APPENDIX A

TABLES DEFINITION AND CREATION

```
CREATE TABLE userdbms(
    user_id     VARCHAR2(25) NOT NULL,
    name        VARCHAR2(50) NOT NULL,
    department  VARCHAR2(25) NOT NULL,
    access_level NUMBER(2)   NOT NULL,
    user_type   VARCHAR2(10) NOT NULL,
    password    VARCHAR2(7)  NOT NULL,
    CONSTRAINT user_user_id_pk PRIMARY KEY(user_id),
    CONSTRAINT user_access_level_ck CHECK
        (access_level IN (01,02,03,04)),
    CONSTRAINT user_user_type_ck CHECK
        (user_type IN ('CUSTOMER','TECHNICIAN',
                'TECHSRVREP','MANAGER')));
```

```
CREATE TABLE customer(
    user_id      VARCHAR2(25)  NOT NULL,
    email        VARCHAR2(50)  NOT NULL,
    address      VARCHAR2(100) NOT NULL,
    phone_no     VARCHAR2(30)  NOT NULL,
    fax_no       VARCHAR2(30)  NOT NULL,
    techsrvrep_id VARCHAR2(25)  NOT NULL,
    CONSTRAINT customer_user_id_pk PRIMARY KEY(user_id),
    CONSTRAINT customer_user_id_fk FOREIGN KEY(user_id)
        REFERENCES userdbms(user_id),
    CONSTRAINT customer_techsrvrep_id_fk FOREIGN KEY(techsrvrep_id)
        REFERENCES techsrvrep(user_id));
```

```
CREATE TABLE technician(
        user_id VARCHAR2(25) NOT NULL,
        CONSTRAINT technician_user_id_pk PRIMARY KEY(user_id),
        CONSTRAINT technician_user_id_fk FOREIGN KEY (user_id)
            REFERENCES userdbms (user_id));
```

```
CREATE TABLE techsrvrep(
    user_id   VARCHAR2(25) NOT NULL,
    email     VARCHAR2(50) NOT NULL,
    address   VARCHAR2(100) NOT NULL,
    phone_no  VARCHAR2(30) NOT NULL,
    fax_no    VARCHAR2(30) NOT NULL,
    CONSTRAINT techsrvrep_user_id_pk PRIMARY KEY(user_id),
    CONSTRAINT techsrvrep_user_id_fk FOREIGN KEY(user_id)
        REFERENCES userdbms(user_id));
```

```
CREATE TABLE manager(
    user_id VARCHAR2(25) NOT NULL,
    email   VARCHAR2(50) NOT NULL,
    address VARCHAR2(100) NOT NULL,
    phone_no VARCHAR2(30) NOT NULL,
    fax_no  VARCHAR2(30) NOT NULL,
    CONSTRAINT manager_user_id_pk PRIMARY KEY(user_id),
    CONSTRAINT manager_user_id_fk FOREIGN KEY(user_id)
        REFERENCES userdbms(user_id));
```

```
CREATE TABLE request(
    request_id NUMBER(6) NOT NULL,
    sample_id  VARCHAR2(15) NOT NULL,
    construction VARCHAR2(100) NOT NULL,
    workorder_numbers_affected VARCHAR2(100) NOT NULL,
    other_layered_materials VARCHAR2(100) NOT NULL,
    mfg_date   DATE NOT NULL,
    technician_id  VARCHAR2(25),
    nature_ofthe_problem VARCHAR2(400) NOT NULL,
    lab_support_requested VARCHAR2(100) NOT NULL,
    customer_process_condition VARCHAR2(300) NOT NULL,
    material_type VARCHAR2(16) NOT NULL,
    customer_id VARCHAR2(25) NOT NULL,
    techsrvrep_id VARCHAR2(25) NOT NULL,
    testing_status VARCHAR2(30) NOT NULL,
    date_requested DATE NOT NULL,
    CONSTRAINT request_request_id_pk PRIMARY KEY(request_id),
    CONSTRAINT request_customer_id_fk FOREIGN KEY(customer_id)
        REFERENCES customer(user_id),
    CONSTRAINT request_techsrvrep_id_fk FOREIGN KEY(techsrvrep_id)
        REFERENCES techsrvrep(user_id),
    CONSTRAINT requet_lab_technician_id_fk FOREIGN KEY(technician_id)
        REFERENCES technician(user_id),
    CONSTRAINT request_material_type_ck CHECK
        (material_type IN('MULTILAYER-BOARD','LAMINATE')));
```

```
CREATE TABLE final_request(
        request_id NUMBER(6) NOT NULL,
        propertesting_toperform VARCHAR(100) NOT NULL,
        observations VARCHAR2(400) NOT NULL,
        techsrvrep_id VARCHAR2(25) NOT NULL,
        CONSTRAINT final_request_request_id_pk PRIMARY KEY(request_id),
        CONSTRAINT final_request_request_id_fk FOREIGN KEY(request_id)
        REFERENCES request(request_id),
        CONSTRAINT final_request_techsrvrep_id_fk FOREIGN KEY(techsrvrep_id)
        REFERENCES techsrvrep(user_id));
```

```
CREATE TABLE historical(
        historical_id NUMBER(6) NOT NULL,
        diagnosis VARCHAR2(400) ,
        technical_advice  VARCHAR2(400),
        final_technical_advice VARCHAR2(400),
        manager_id  VARCHAR2(25) NOT NULL,
        CONSTRAINT historical_historical_id_pk PRIMARY KEY(historical_id),
        CONSTRAINT historical_historical_id_fk FOREIGN KEY(historical_id)
        REFERENCES final_request(request_id),
        CONSTRAINT historical_manager_id_fk FOREIGN KEY(manager_id)
        REFERENCES manager(user_id));
```

```
CREATE TABLE testing(
    testing_id NUMBER(8) NOT NULL,
    historical_id NUMBER(6) NOT NULL,
    testingdate DATE ,
    test_type VARCHAR2(10) NOT NULL,
    technician_id VARCHAR2(25) NOT NULL,
    CONSTRAINT testing_id_pk PRIMARY KEY(testing_id),
    CONSTRAINT testing_test_type_ck CHECK
        (test_type IN('TRANSITION','RESISTANCE','RESIDUAL')),
    CONSTRAINT historical_id_fk FOREIGN KEY(historical_id)
        REFERENCES historical(historical_id),
    CONSTRAINT testing_technician_id_fk FOREIGN KEY(technician_id)
        REFERENCES technician
```

```
CREATE TABLE transition(
    testing_id NUMBER(8) NOT NULL,
    tg NUMBER(6,2),
    cte_x NUMBER(6,3),
    cte_y NUMBER(6,3),
    cte_z NUMBER(6,3),
    expansion NUMBER(5,2),
    CONSTRAINT transition_test_id_pk PRIMARY KEY(testing_id),
    CONSTRAINT transition_test_id_fk FOREIGN KEY(testing_id)
    REFERENCES testing(testing_id));
```

```
CREATE TABLE resistance(
    testing_id NUMBER(8) NOT NULL,
    t300 VARCHAR2(4),
    t288 VARCHAR2(4),
    t260 VARCHAR2(4),
    t300_min NUMBER(2),
    t288_min NUMBER(2),
    t260_min NUMBER(2),
    CONSTRAINT resistance_test_id_pk PRIMARY KEY(testing_id),
    CONSTRAINT resistance_test_id_fk FOREIGN KEY(testing_id)
    REFERENCES testing(testing_id));
```

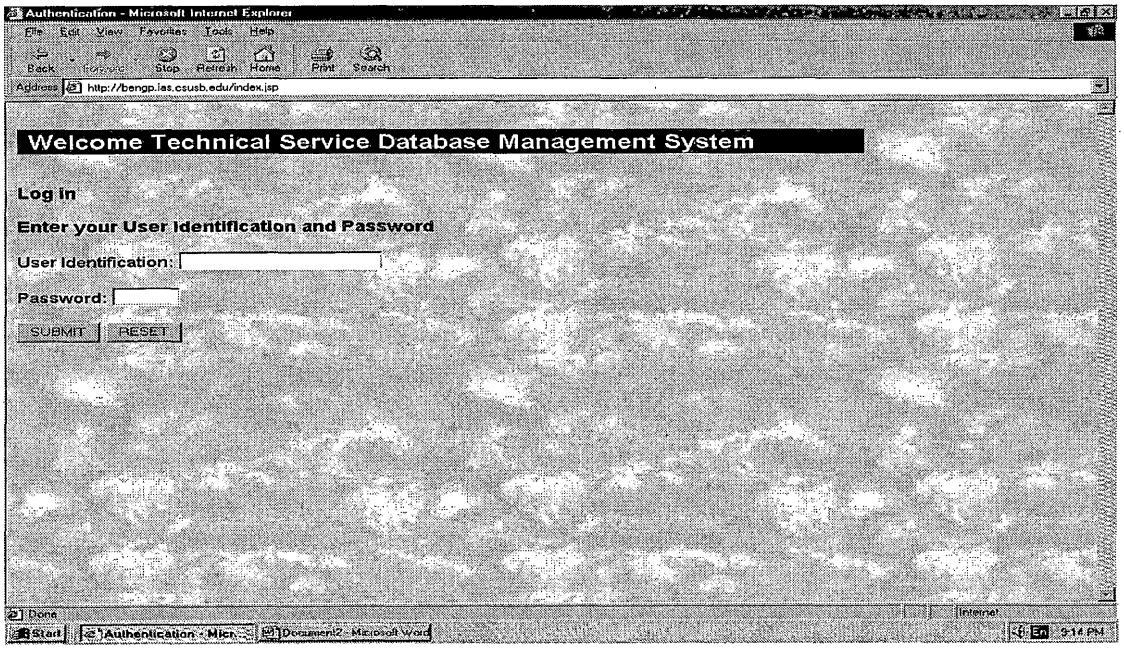
```
CREATE TABLE residual_cure(
    testing_id NUMBER(8) NOT NULL,
    enthalpy NUMBER(6,2),
    CONSTRAINT residual_test_id_pk PRIMARY KEY(testing_id),
    CONSTRAINT residual_test_id_fk FOREIGN KEY(testing_id)
    REFERENCES testing(testing_id));
```

APPENDIX B

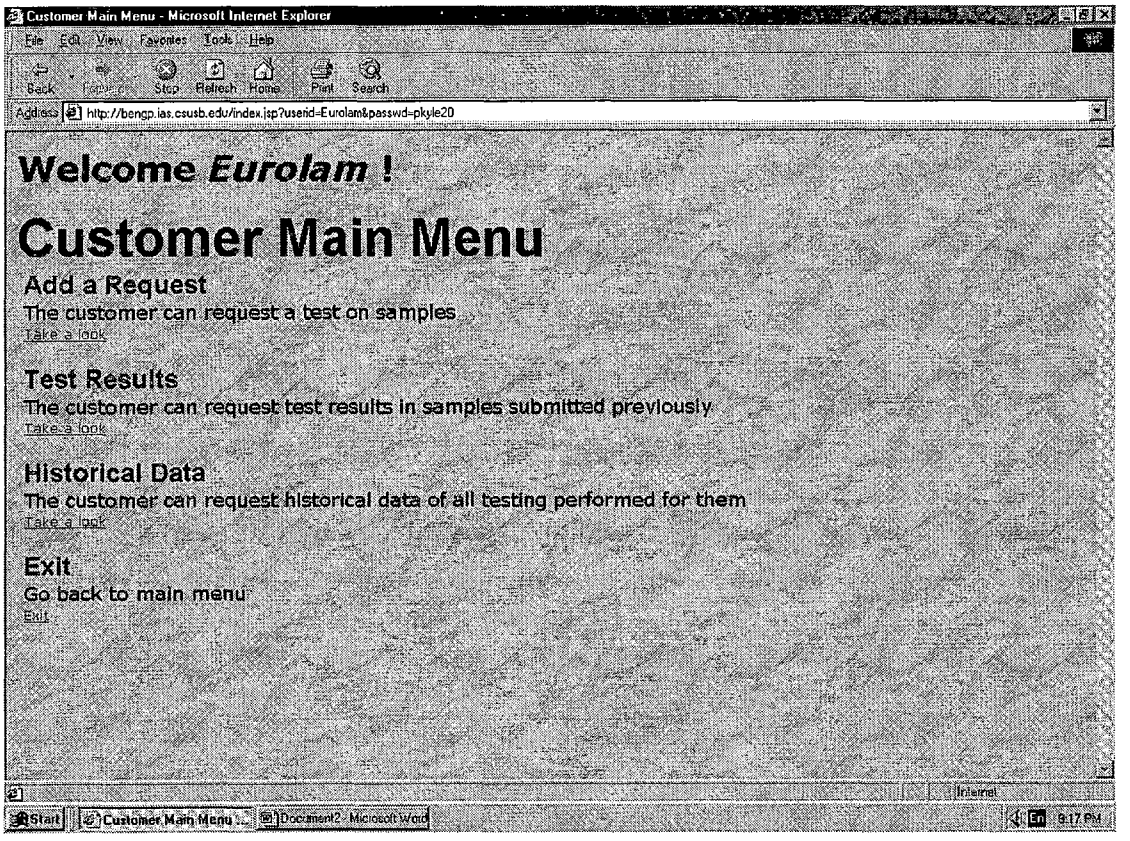
HOME PAGE AND CUSTOMER

KEY PATH

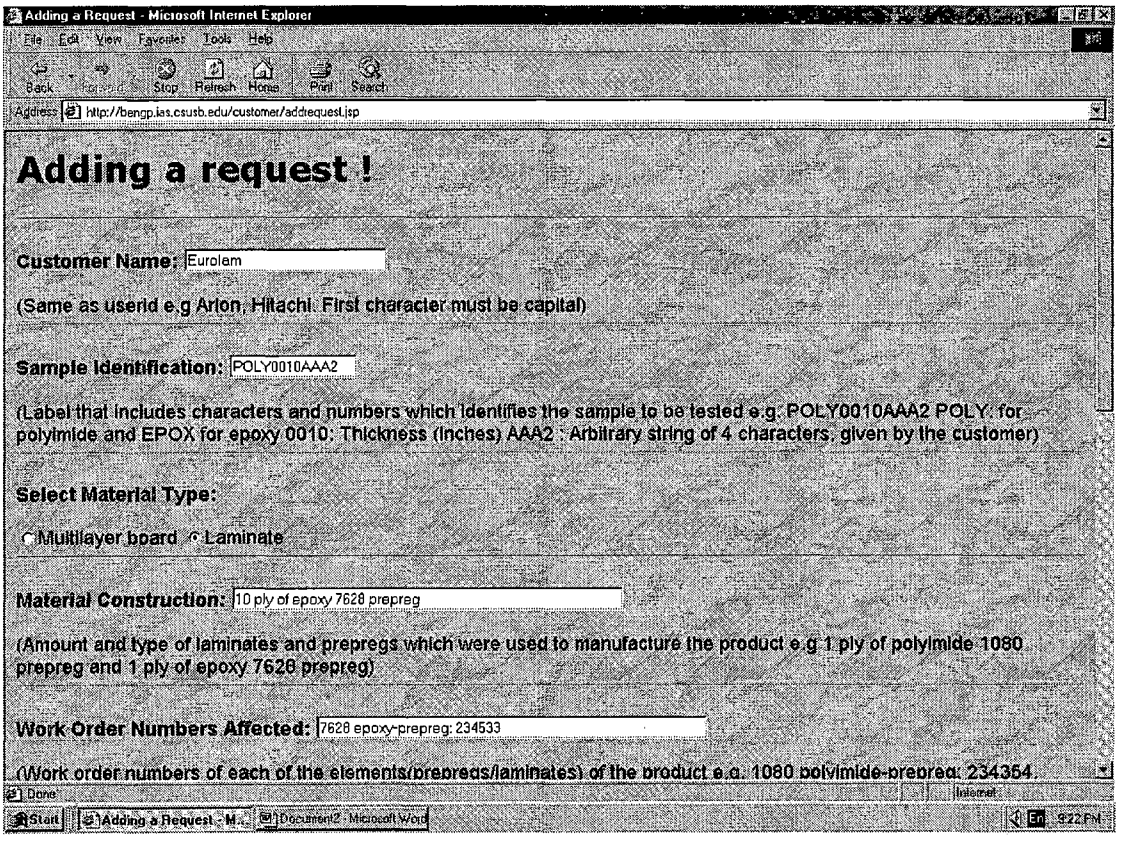
HOME


Authentication - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Print Search
Address http://bengp.las.csusb.edu/index.jsp
Welcome Technical Service Database Management System
Log in
Enter your User Identification and Password
User Identification:
Password:
SUBMIT
RESET
Done
Internet
Start
Authentication - Micr...
Document2 - Microsoft Word
9:14 PM

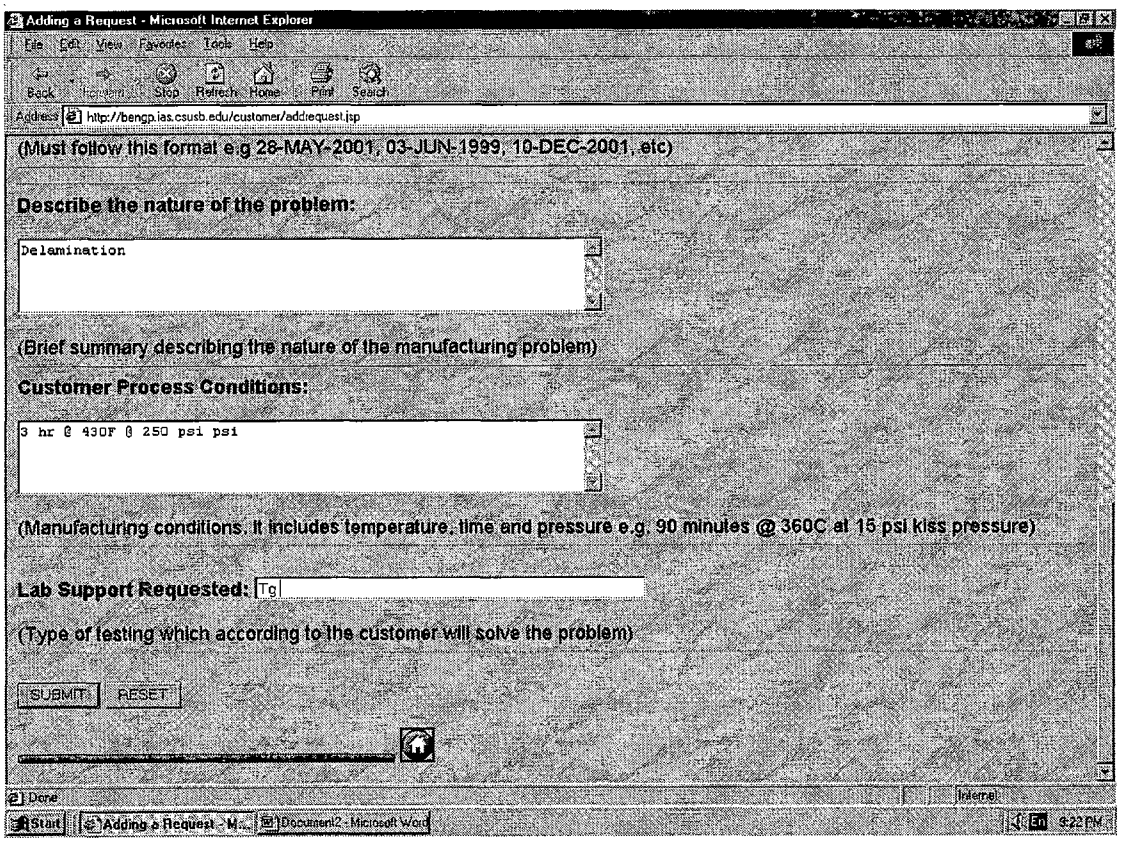
CUSTOMER MAIN MENU



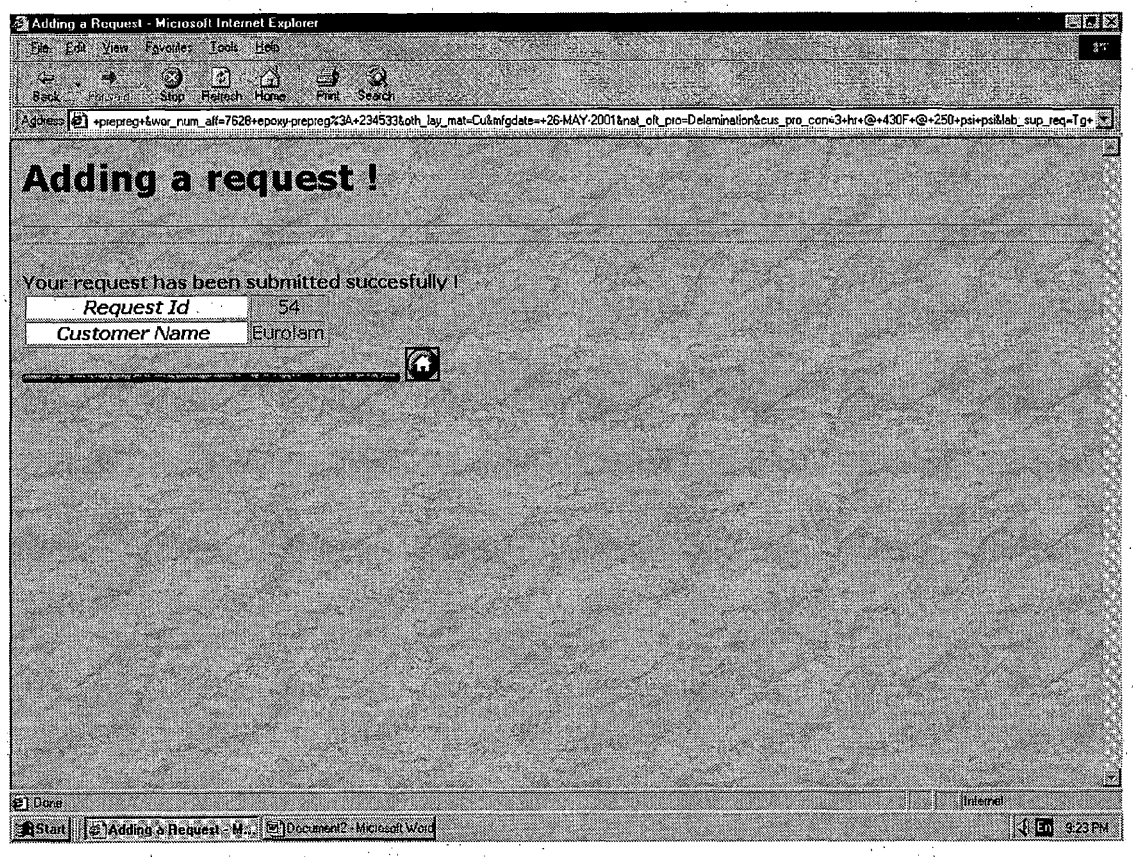
ADD A REQUEST



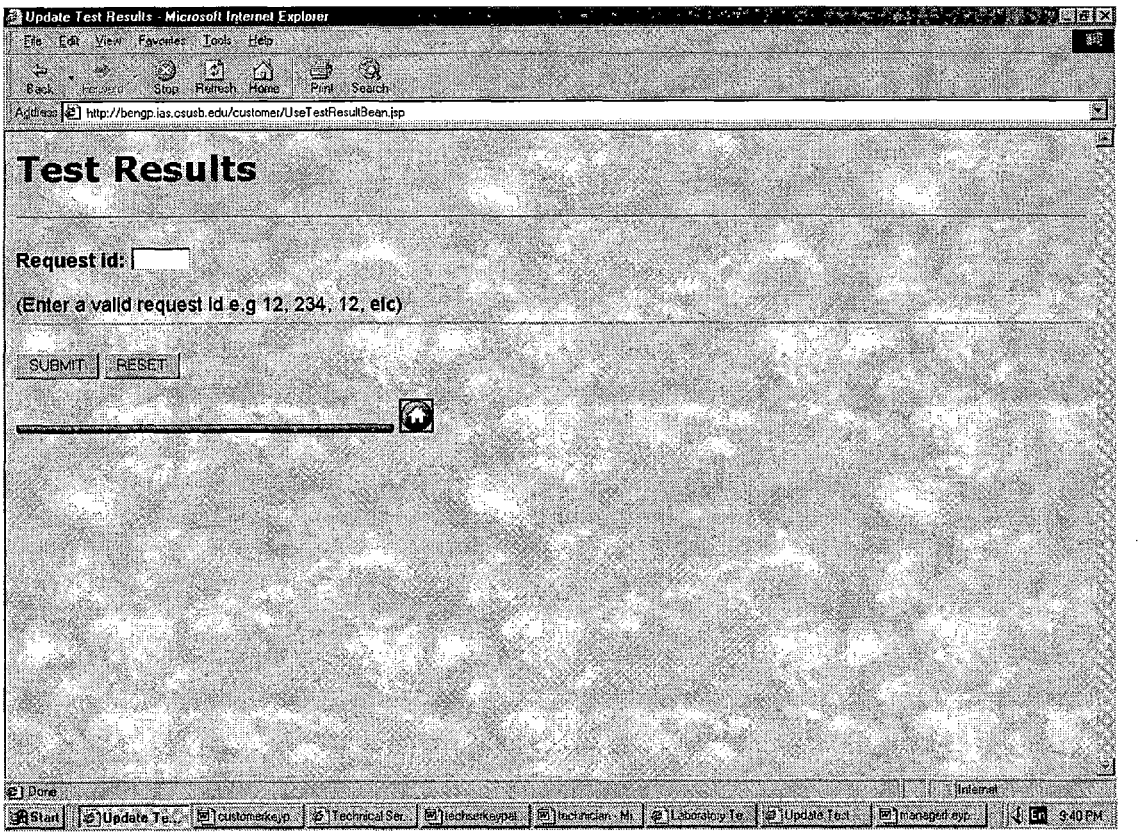
ADD A REQUEST



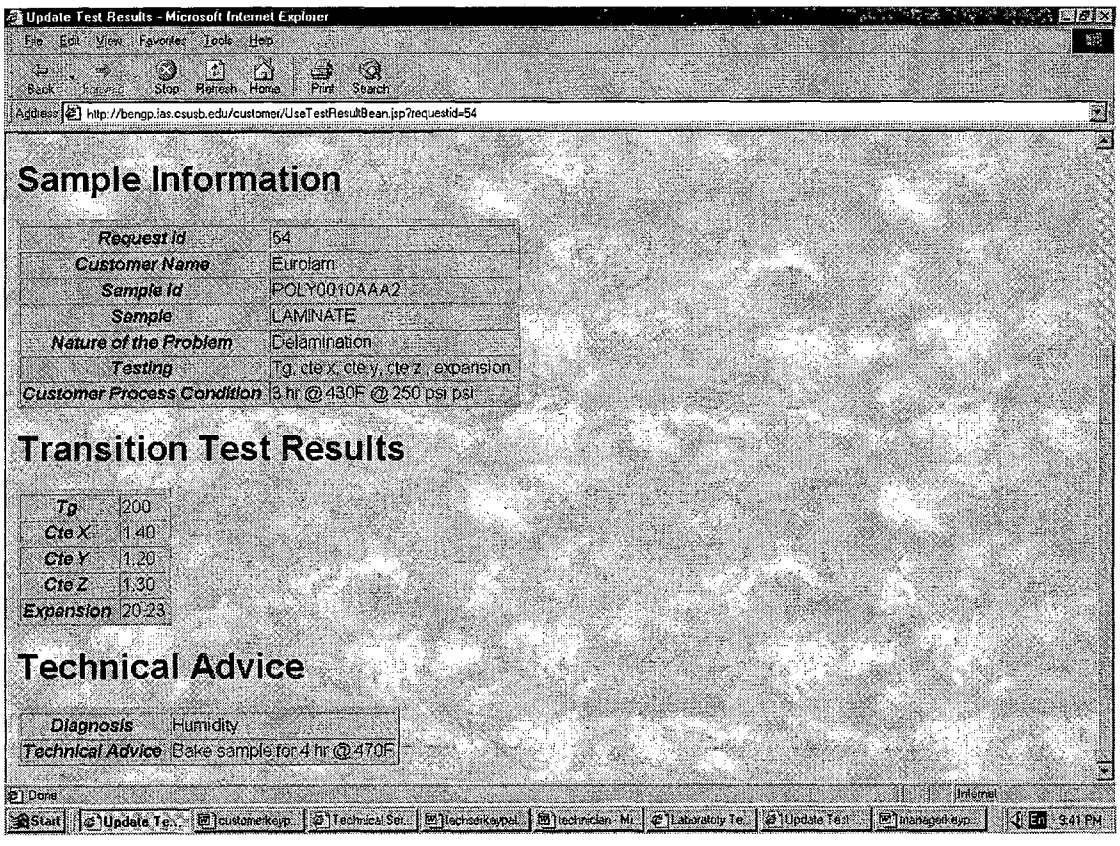
ADD A REQUEST



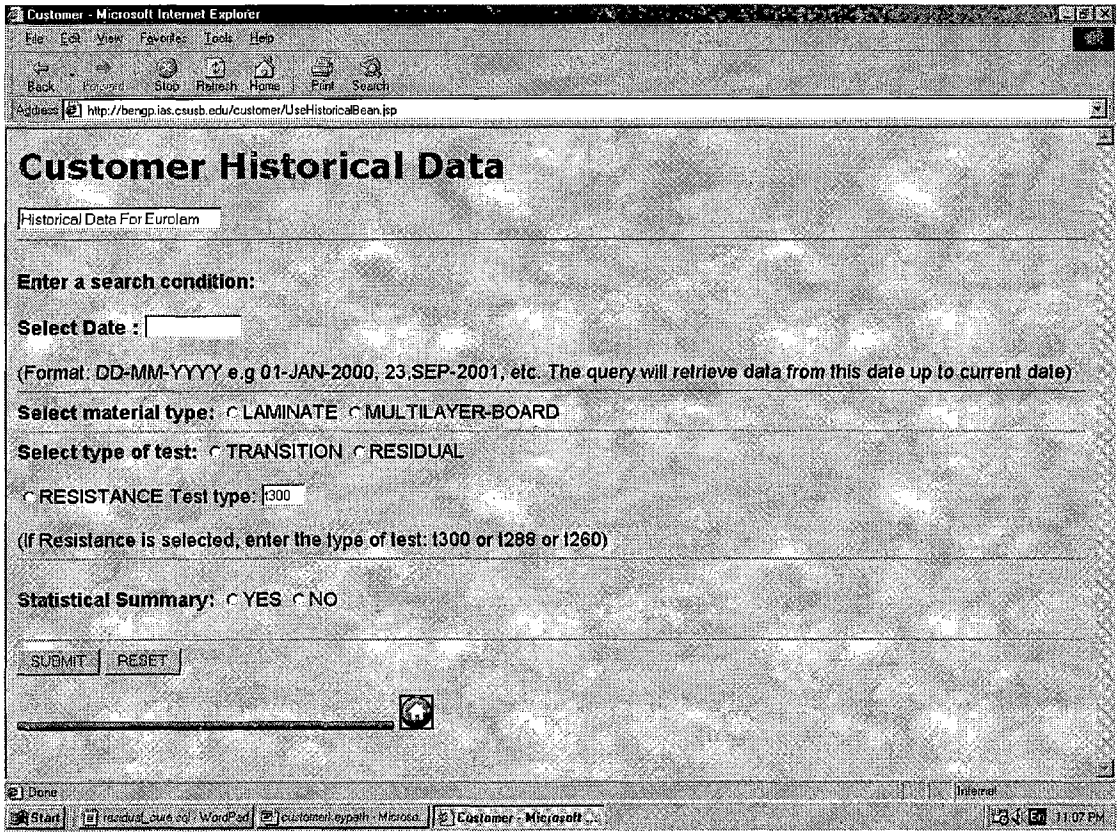
TEST RESULTS



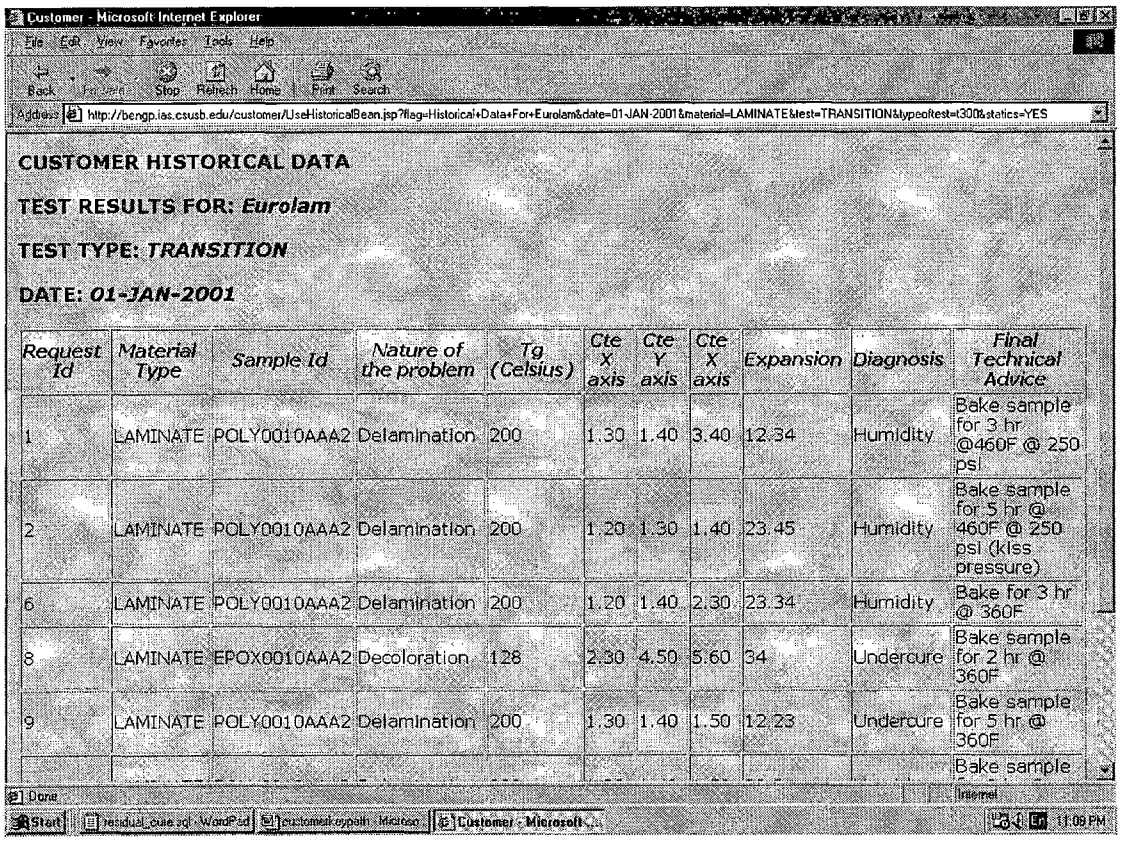
TEST RESULTS



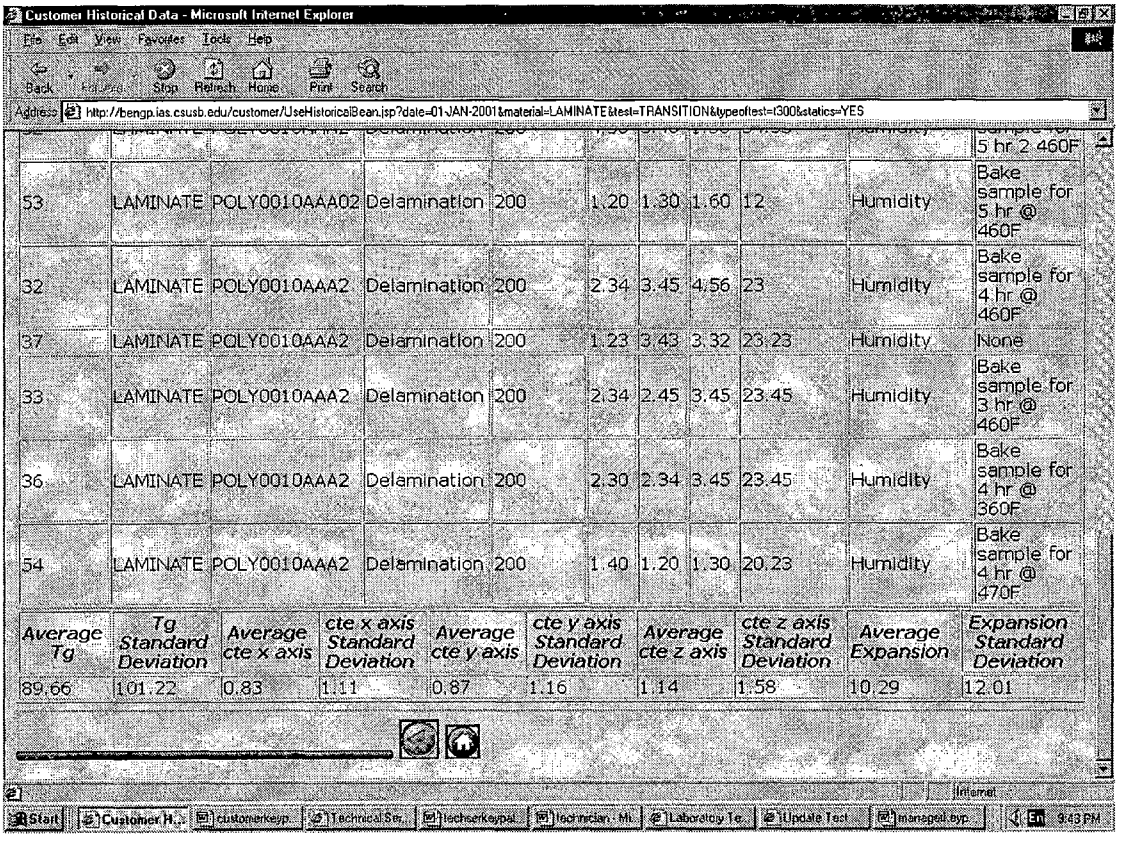
HISTORICAL DATA



DISPLAY HISTORICAL DATA



DISPLAY HISTORICAL DATA



										5 hr 2 460F
53	LAMINATE	POLY0010AAA02	Delamination	200	1.20	1.30	1.60	12	Humidity	Bake sample for 5 hr @ 460F
32	LAMINATE	POLY0010AAA2	Delamination	200	2.34	3.45	4.56	23	Humidity	Bake sample for 4 hr @ 460F
37	LAMINATE	POLY0010AAA2	Delamination	200	1.23	3.43	3.32	23.23	Humidity	None
33	LAMINATE	POLY0010AAA2	Delamination	200	2.34	2.45	3.45	23.45	Humidity	Bake sample for 3 hr @ 460F
36	LAMINATE	POLY0010AAA2	Delamination	200	2.30	2.34	3.45	23.45	Humidity	Bake sample for 4 hr @ 360F
54	LAMINATE	POLY0010AAA2	Delamination	200	1.40	1.20	1.30	20.23	Humidity	Bake sample for 4 hr @ 470F

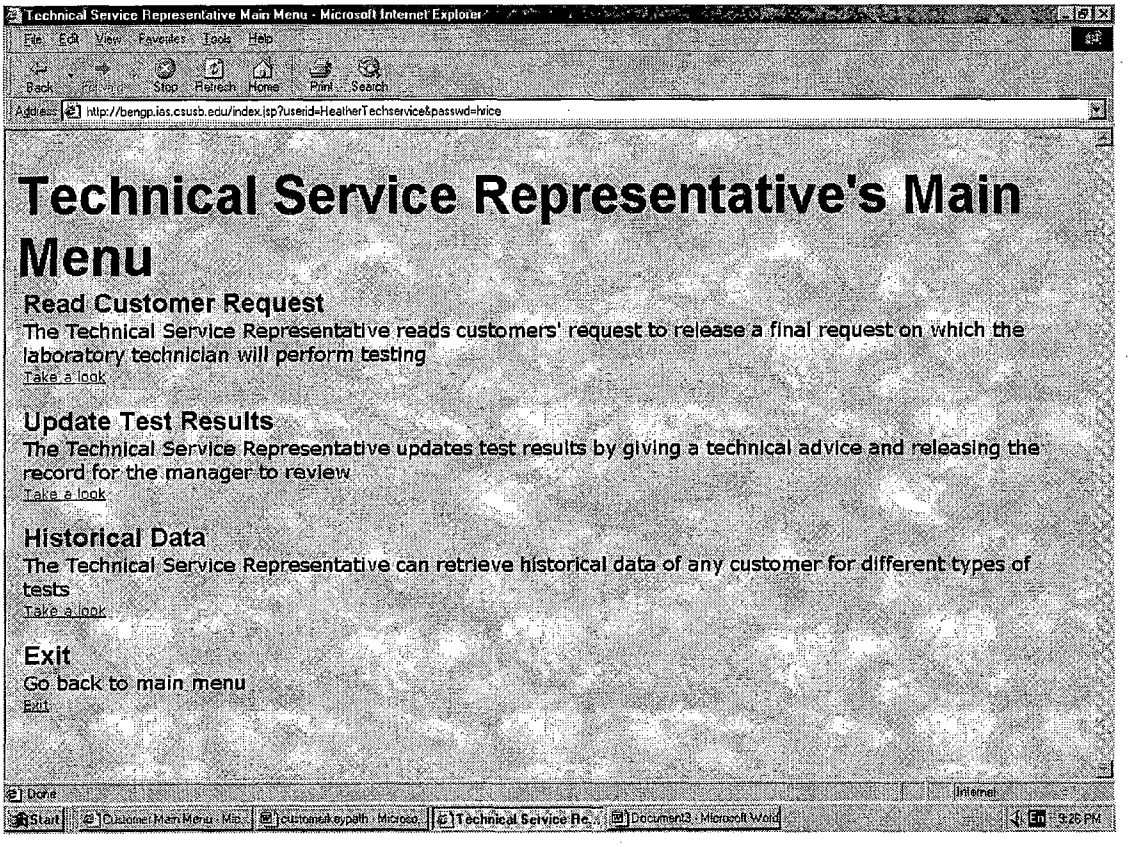
Average Tg	*Tg Standard Deviation*	*Average cte x axis*	*cte x axis Standard Deviation*	*Average cte y axis*	*cte y axis Standard Deviation*	*Average cte z axis*	*cte z axis Standard Deviation*	*Average Expansion*	*Expansion Standard Deviation*
89.66	101.22	0.83	1.11	0.87	1.16	1.14	1.58	10.29	12.01

APPENDIX C

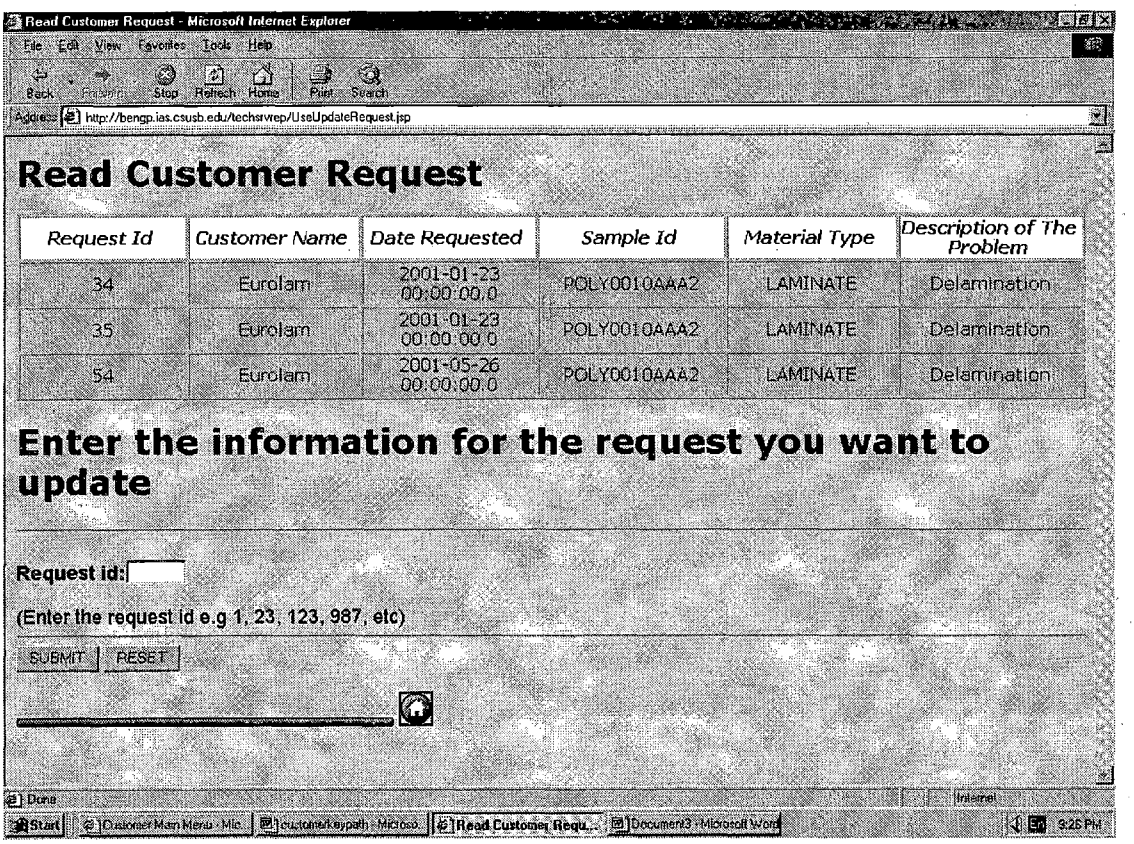
TECHNICAL SERVICE

REPRESENTATIVE KEY PATH

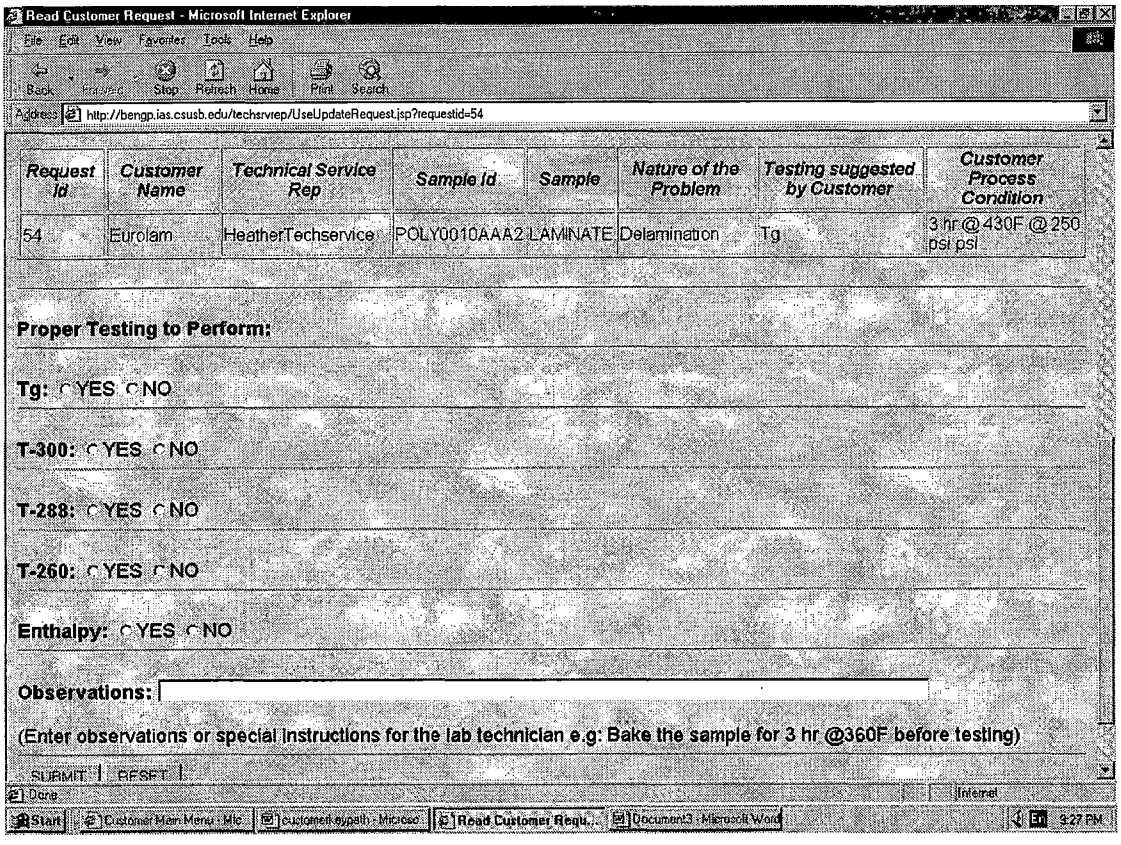
TECHNICAL SERVICE REPRESENTATIVE MAIN MENU



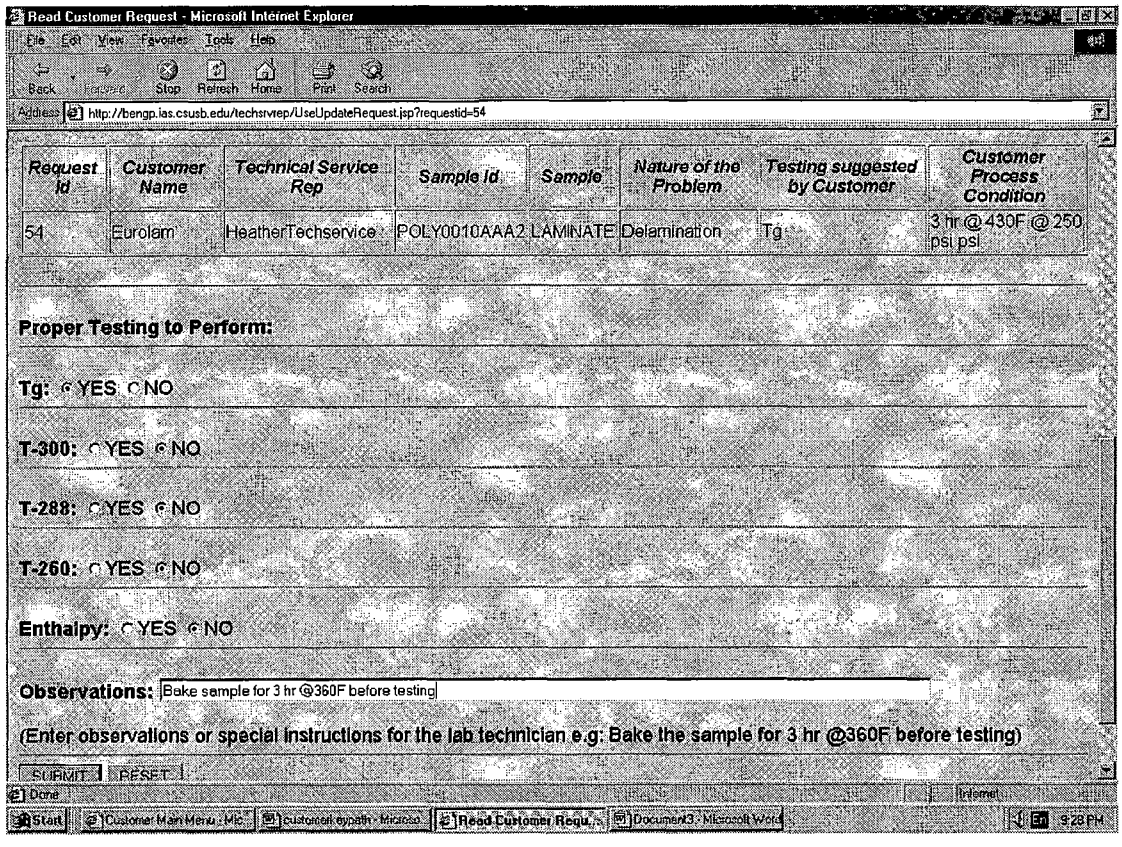
UPDATE CUSTOMER REQUEST



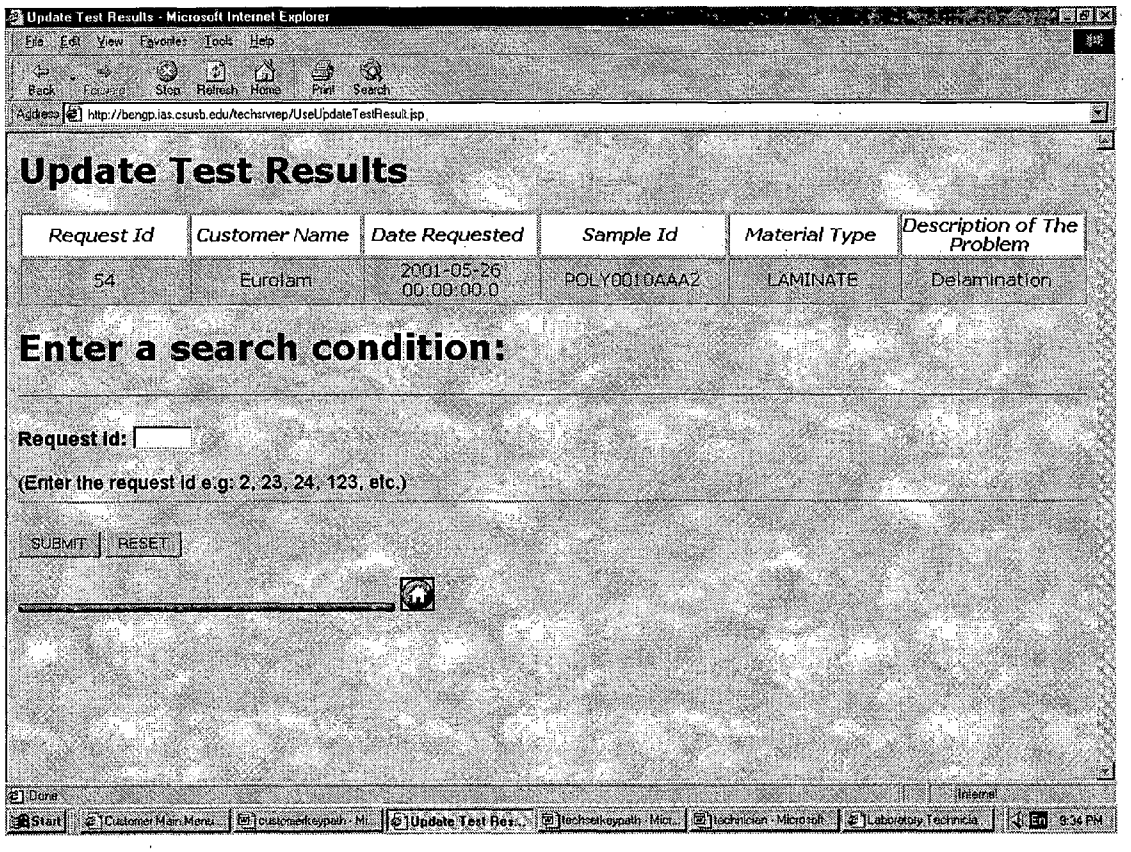
UPDATE CUSTOMER REQUEST



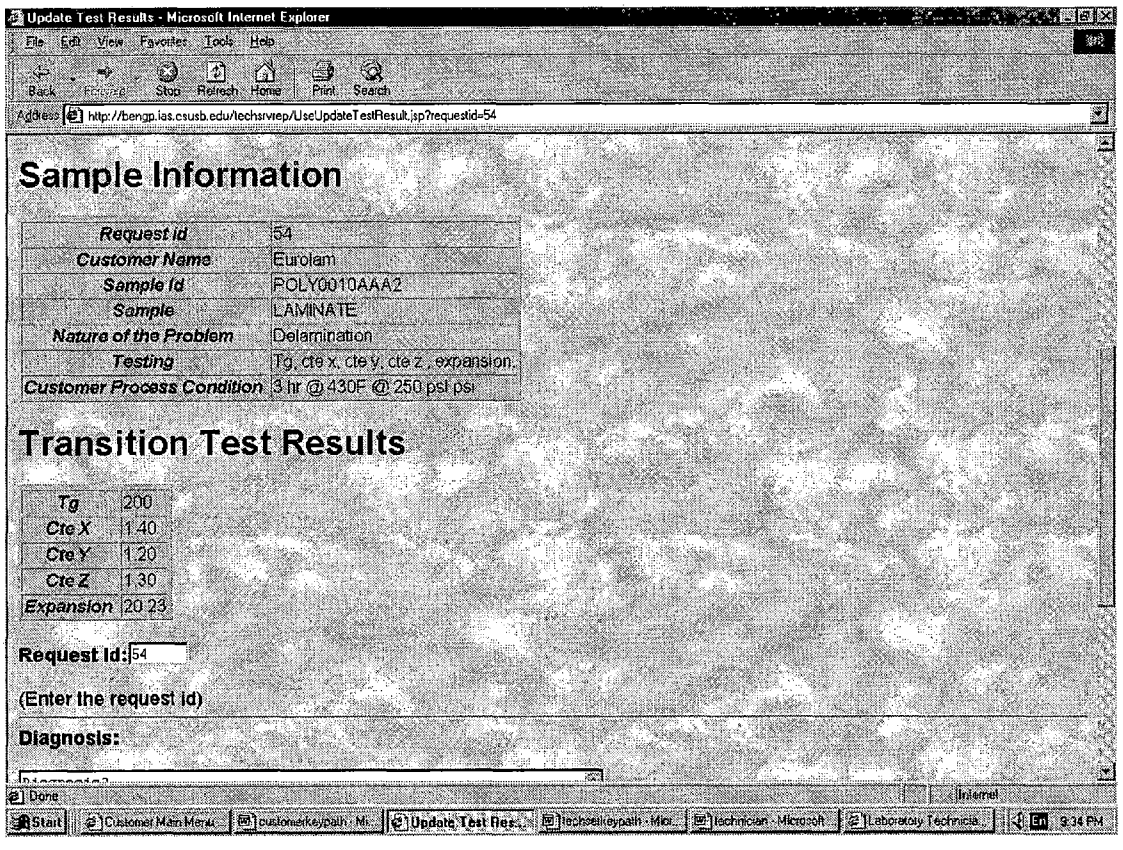
UPDATE CUSTOMER REQUEST



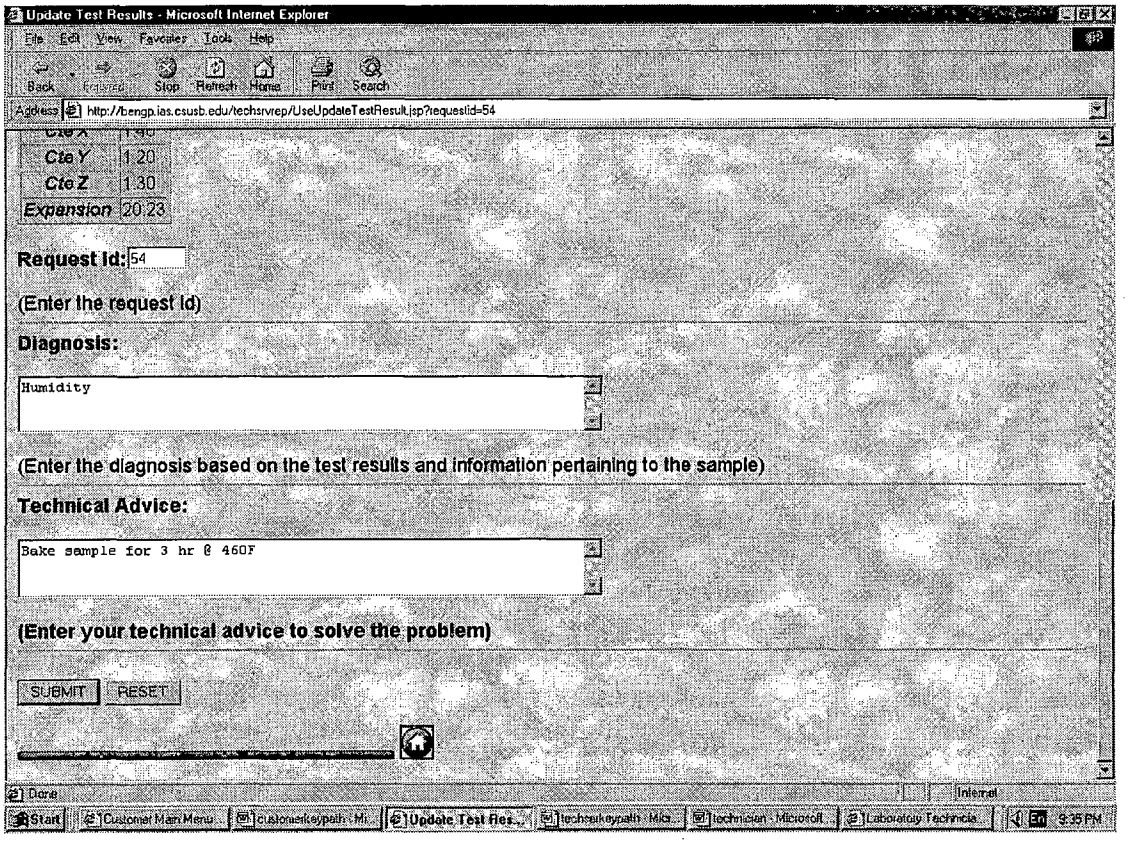
UPDATE TEST RESULTS



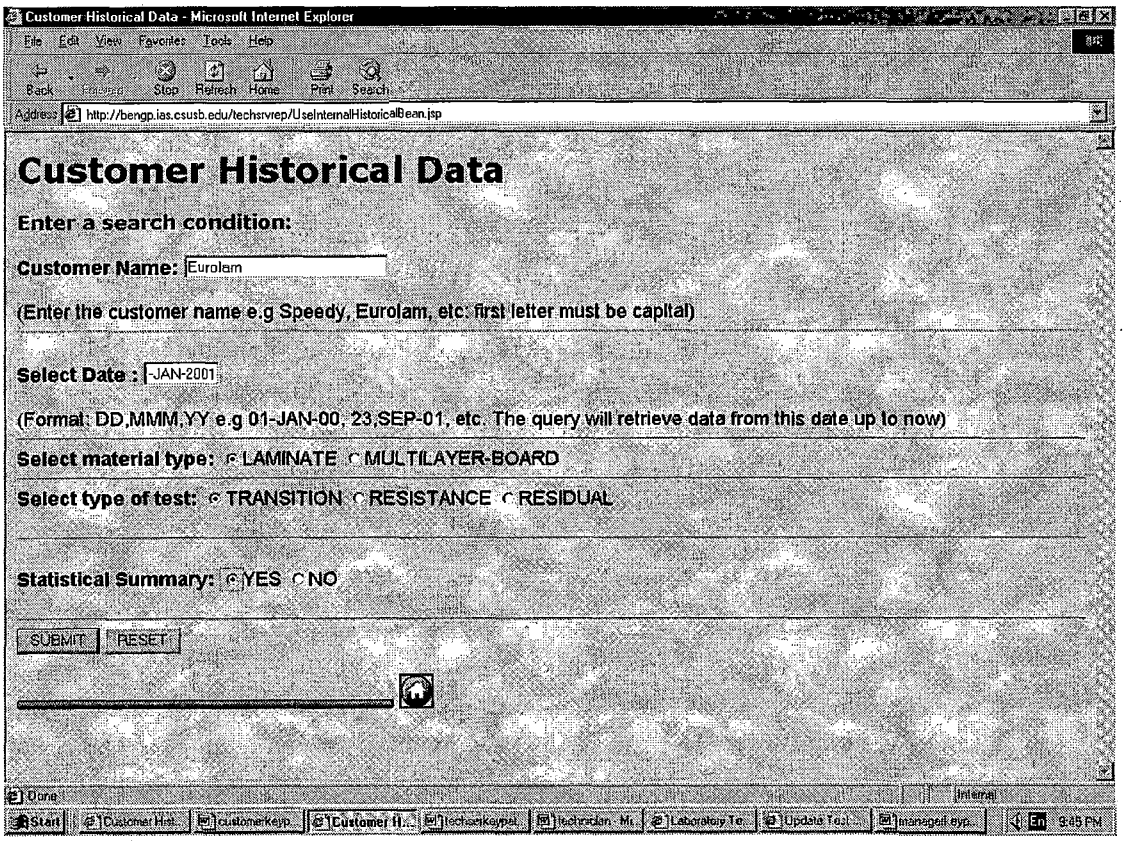
UPDATE TEST RESULTS



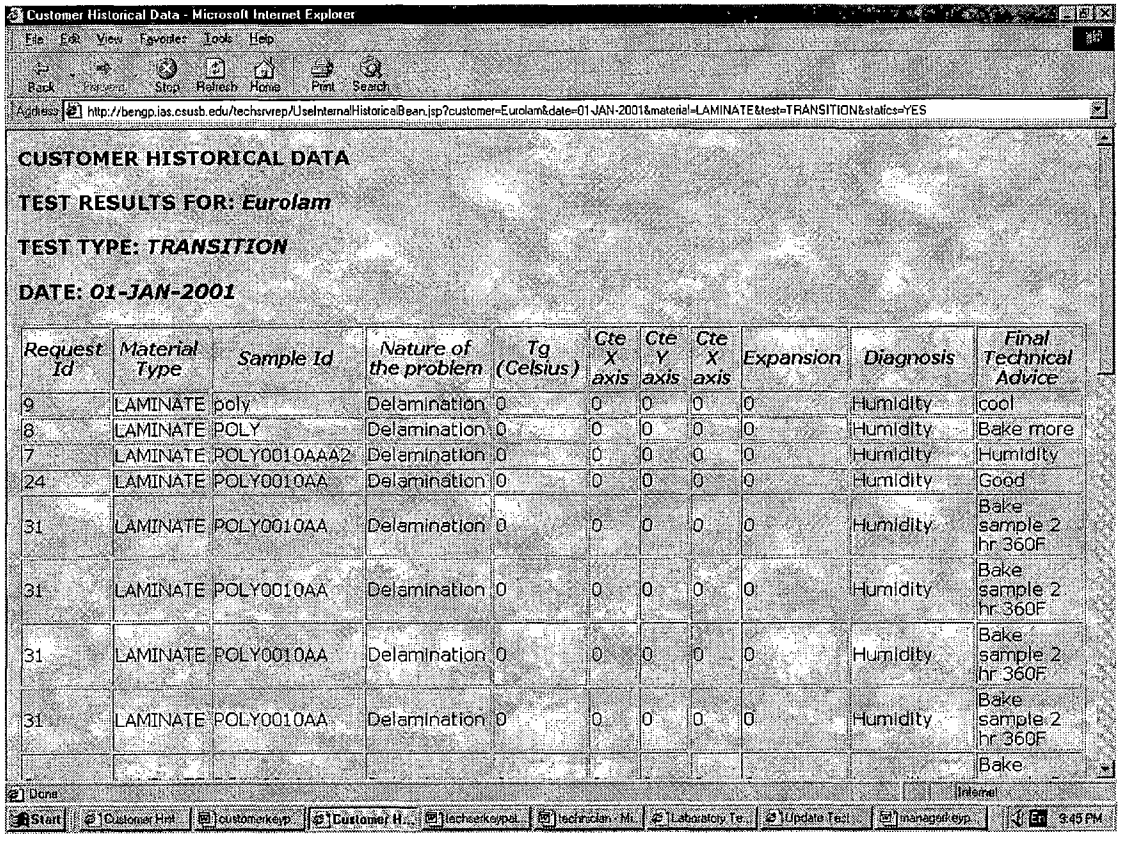
UPDATE TEST RESULTS



HISTORICAL DATA



HISTORICAL DATA



CUSTOMER HISTORICAL DATA

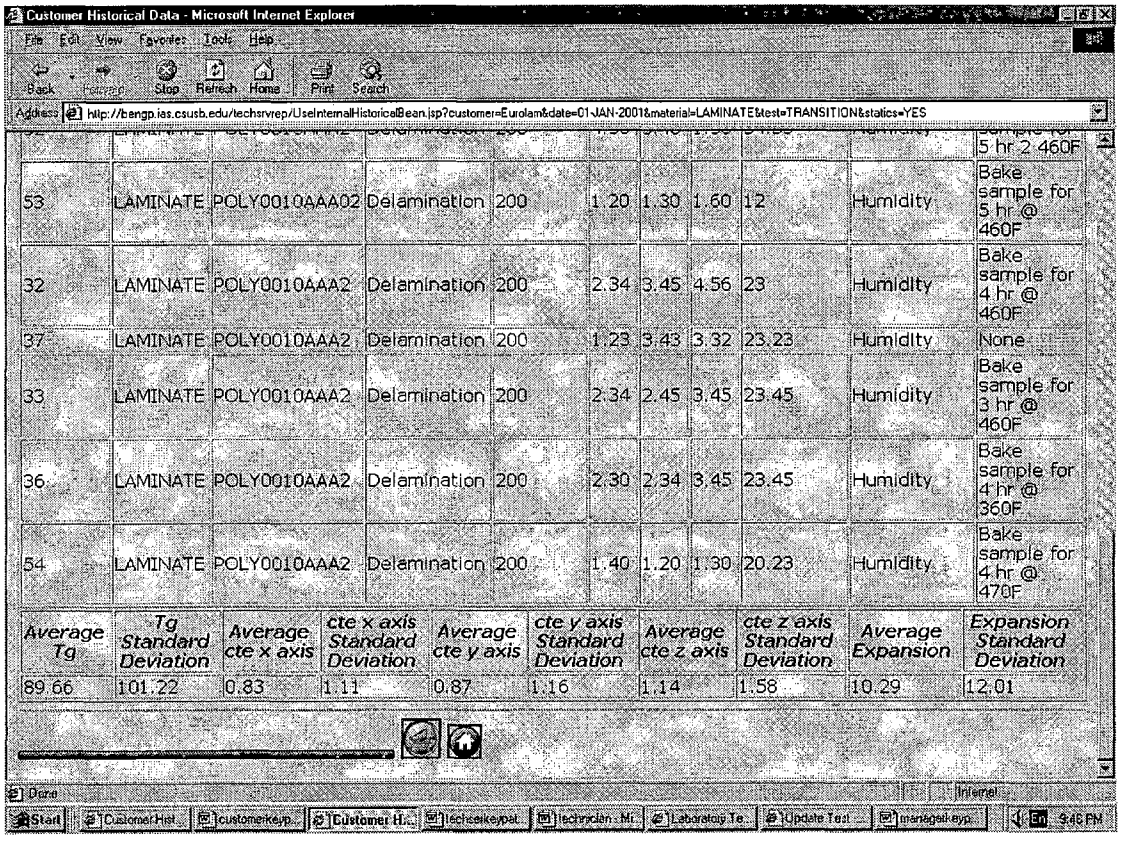
TEST RESULTS FOR: *Eurolam*

TEST TYPE: *TRANSITION*

DATE: *01-JAN-2001*

Request Id	*Material Type*	*Sample Id*	*Nature of the problem*	*Tg (Celsius)*	*Cte X axis*	*Cte Y axis*	*Cte X axis*	*Expansion*	*Diagnosis*	*Final Technical Advice*
9	LAMINATE	poly	Delamination	0	0	0	0	0	Humidity	cool
8	LAMINATE	POLY	Delamination	0	0	0	0	0	Humidity	Bake more
7	LAMINATE	POLY0010AAA2	Delamination	0	0	0	0	0	Humidity	Humidity
24	LAMINATE	POLY0010AA	Delamination	0	0	0	0	0	Humidity	Good
31	LAMINATE	POLY0010AA	Delamination	0	0	0	0	0	Humidity	Bake sample 2 hr 360F
31	LAMINATE	POLY0010AA	Delamination	0	0	0	0	0	Humidity	Bake sample 2 hr 360F
31	LAMINATE	POLY0010AA	Delamination	0	0	0	0	0	Humidity	Bake sample 2 hr 360F
31	LAMINATE	POLY0010AA	Delamination	0	0	0	0	0	Humidity	Bake sample 2 hr 360F
										Bake

HISTORICAL DATA



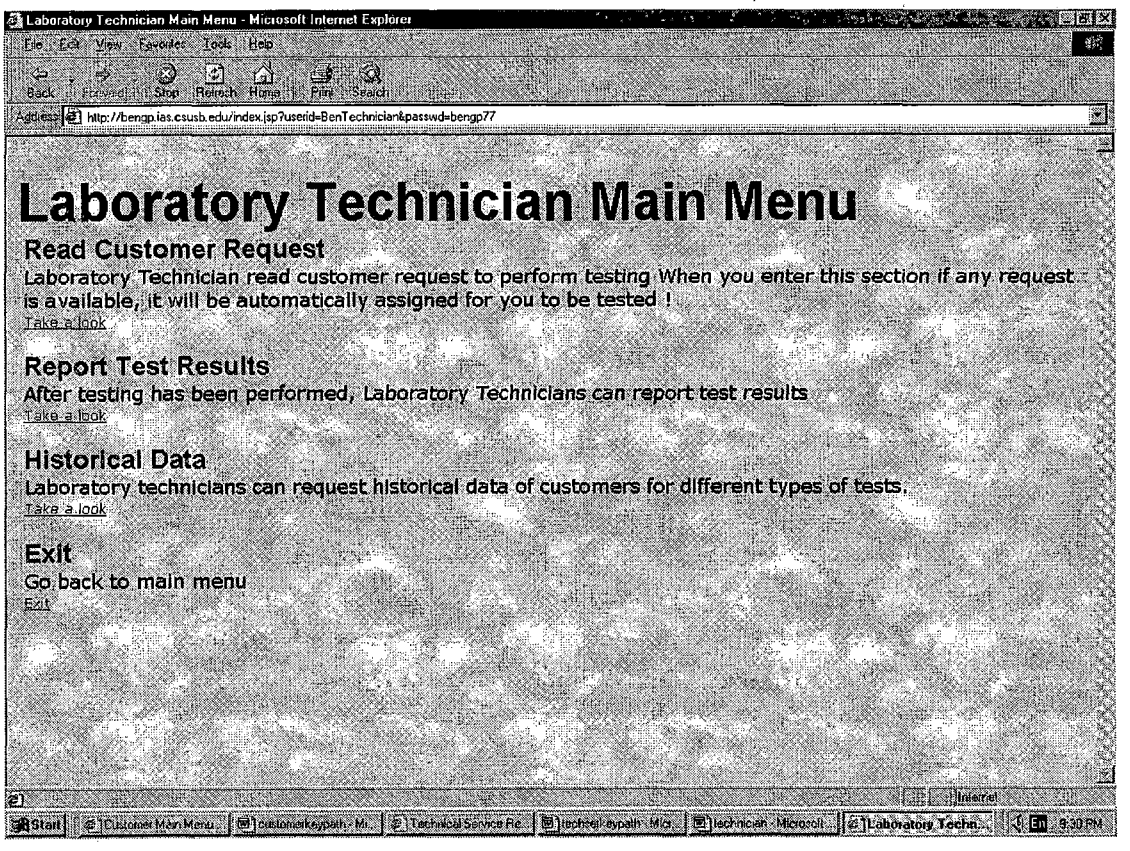
										5 hr 2 460F
53	LAMINATE	POLY0010AAA02	Delamination	200	1.20	1.30	1.60	12	Humidity	Bake sample for 5 hr @ 460F
32	LAMINATE	POLY0010AAA2	Delamination	200	2.34	3.45	4.56	23	Humidity	Bake sample for 4 hr @ 460F
37	LAMINATE	POLY0010AAA2	Delamination	200	1.23	3.43	3.32	23.23	Humidity	None
33	LAMINATE	POLY0010AAA2	Delamination	200	2.34	2.45	3.45	23.45	Humidity	Bake sample for 3 hr @ 460F
36	LAMINATE	POLY0010AAA2	Delamination	200	2.30	2.34	3.45	23.45	Humidity	Bake sample for 4 hr @ 360F
54	LAMINATE	POLY0010AAA2	Delamination	200	1.40	1.20	1.30	20.23	Humidity	Bake sample for 4 hr @ 470F

Average Tg	*Tg Standard Deviation*	*Average cte x axis*	*cte x axis Standard Deviation*	*Average cte y axis*	*cte y axis Standard Deviation*	*Average cte z axis*	*cte z axis Standard Deviation*	*Average Expansion*	*Expansion Standard Deviation*
89.66	101.22	0.83	1.11	0.87	1.16	1.14	1.58	10.29	12.01

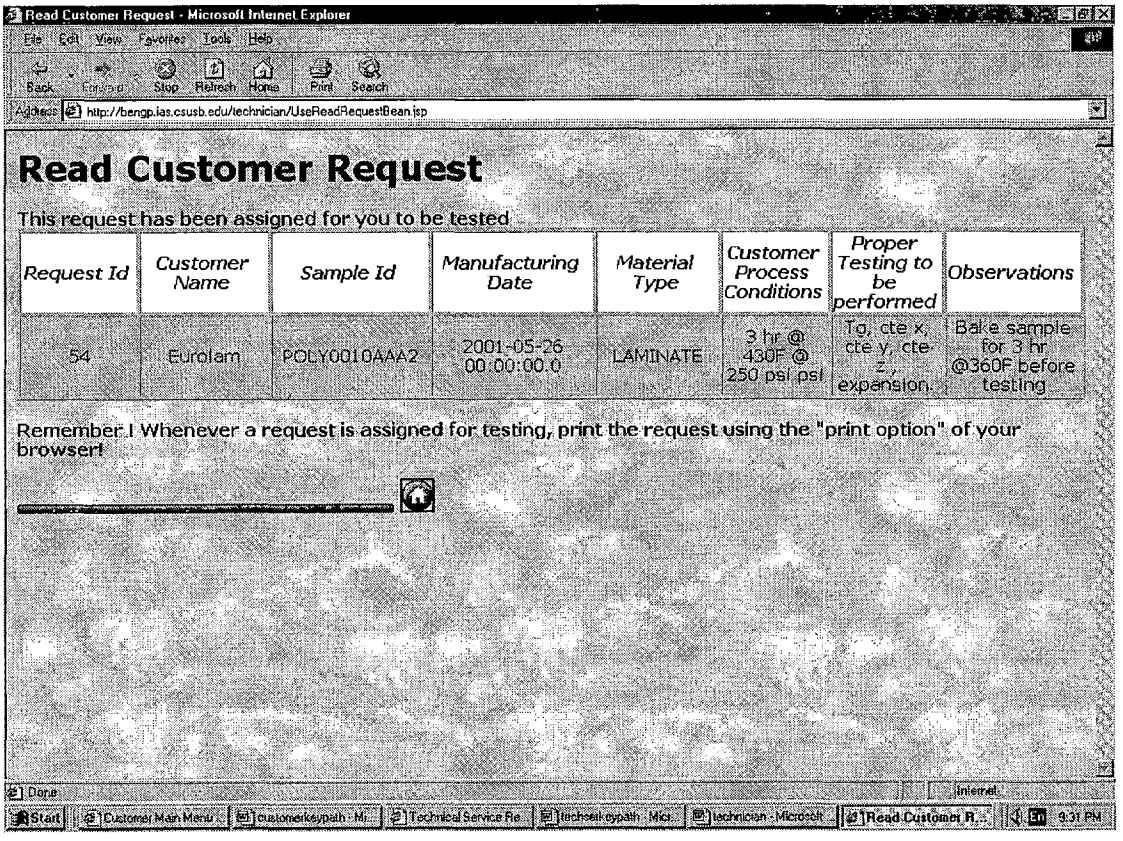
APPENDIX D

LABORATORY TECHNICIAN KEY PATH

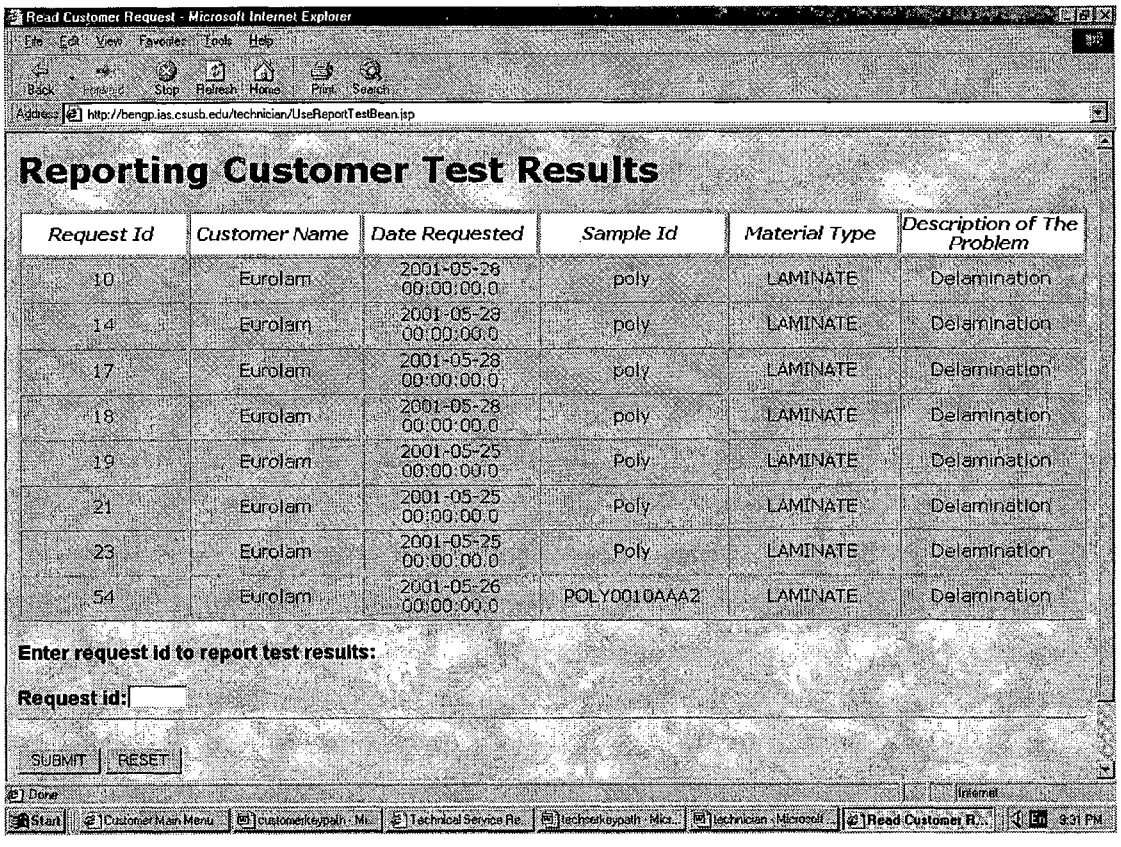
LABORATORY TECHNICIAN MAIN MENU



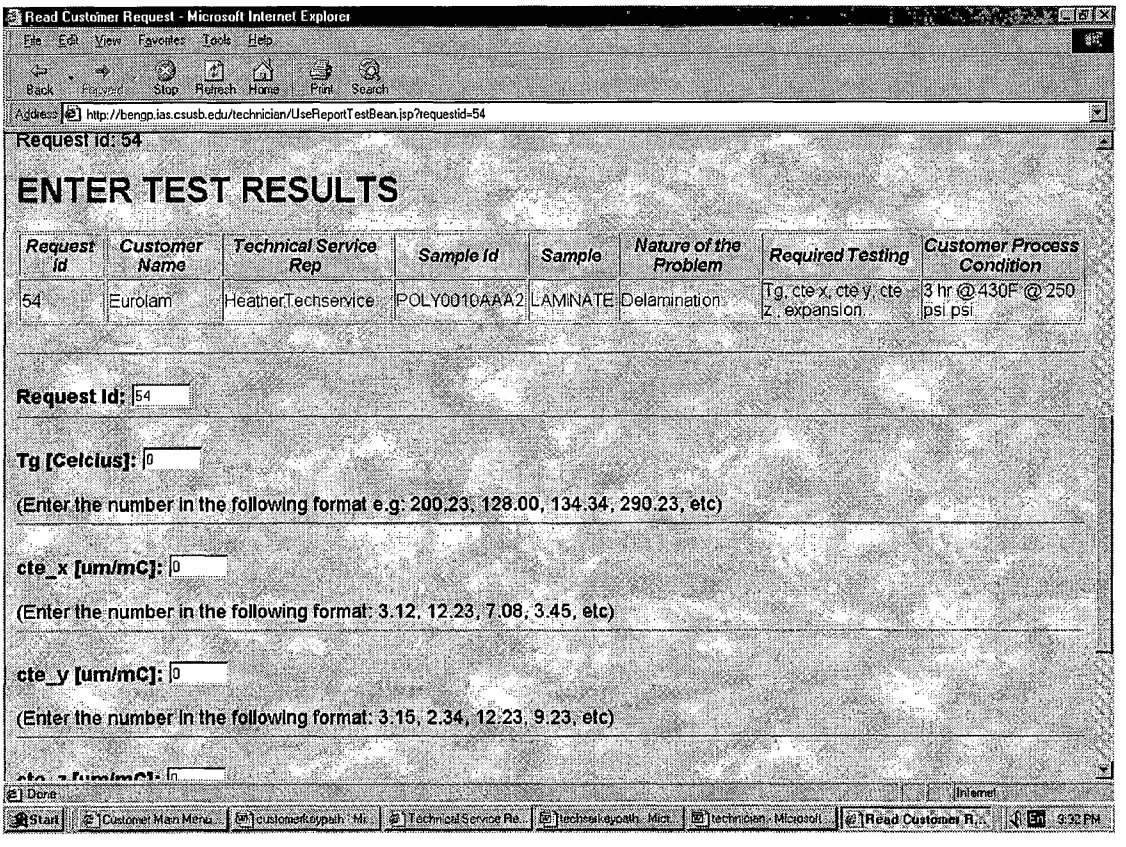
READ CUSTOMER REQUEST



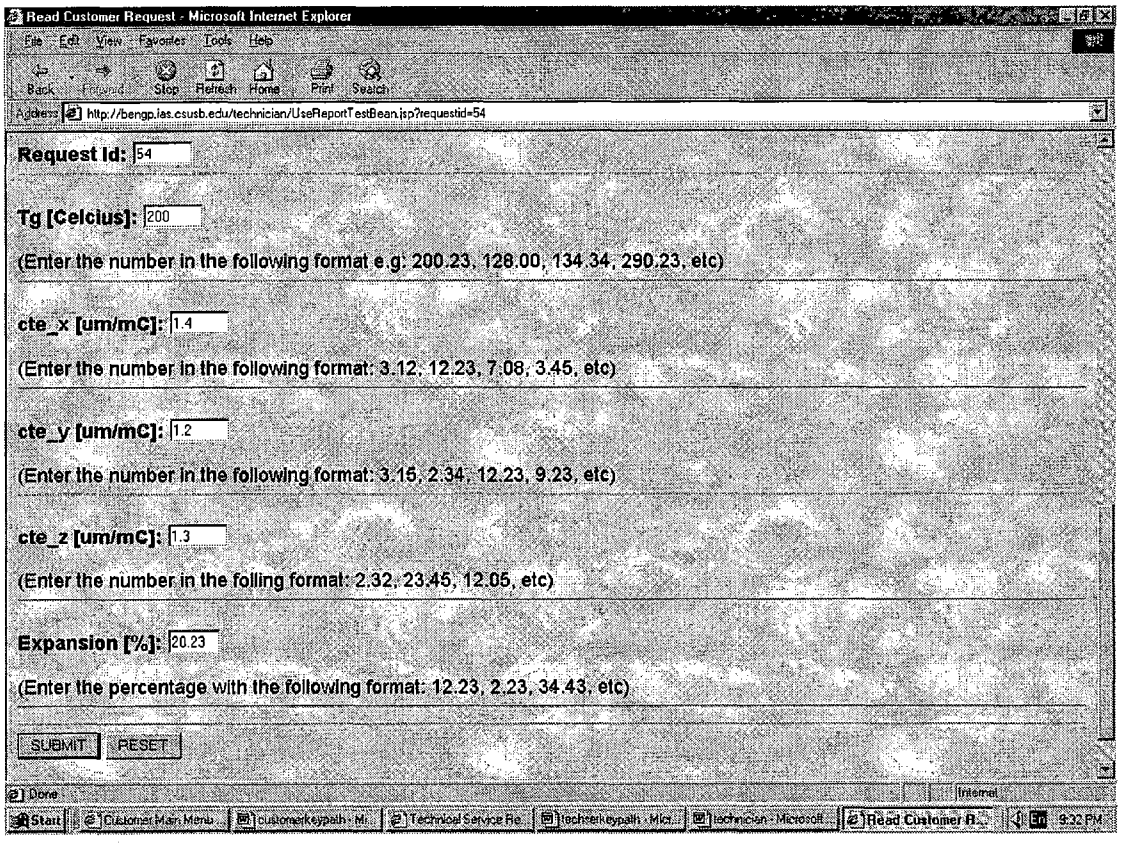
REPORT TEST RESULTS



REPORT TEST RESULTS



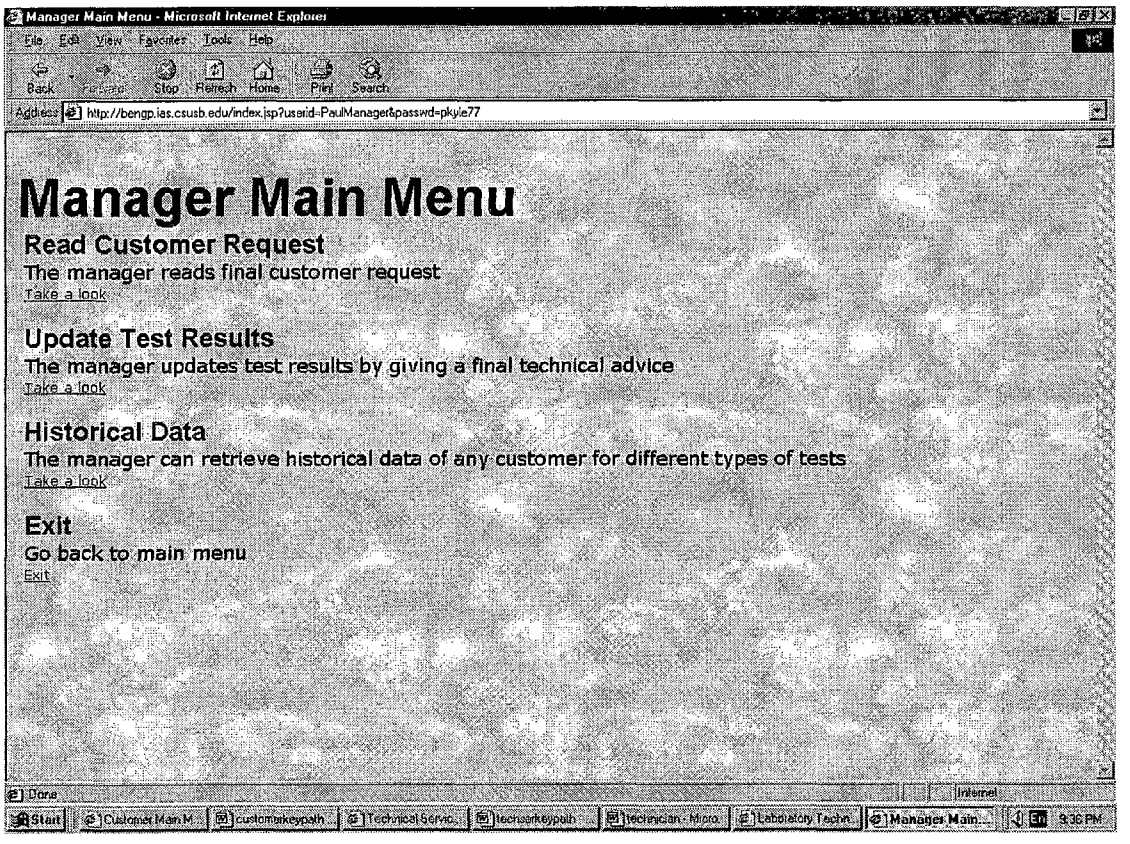
REPORT TEST RESULTS



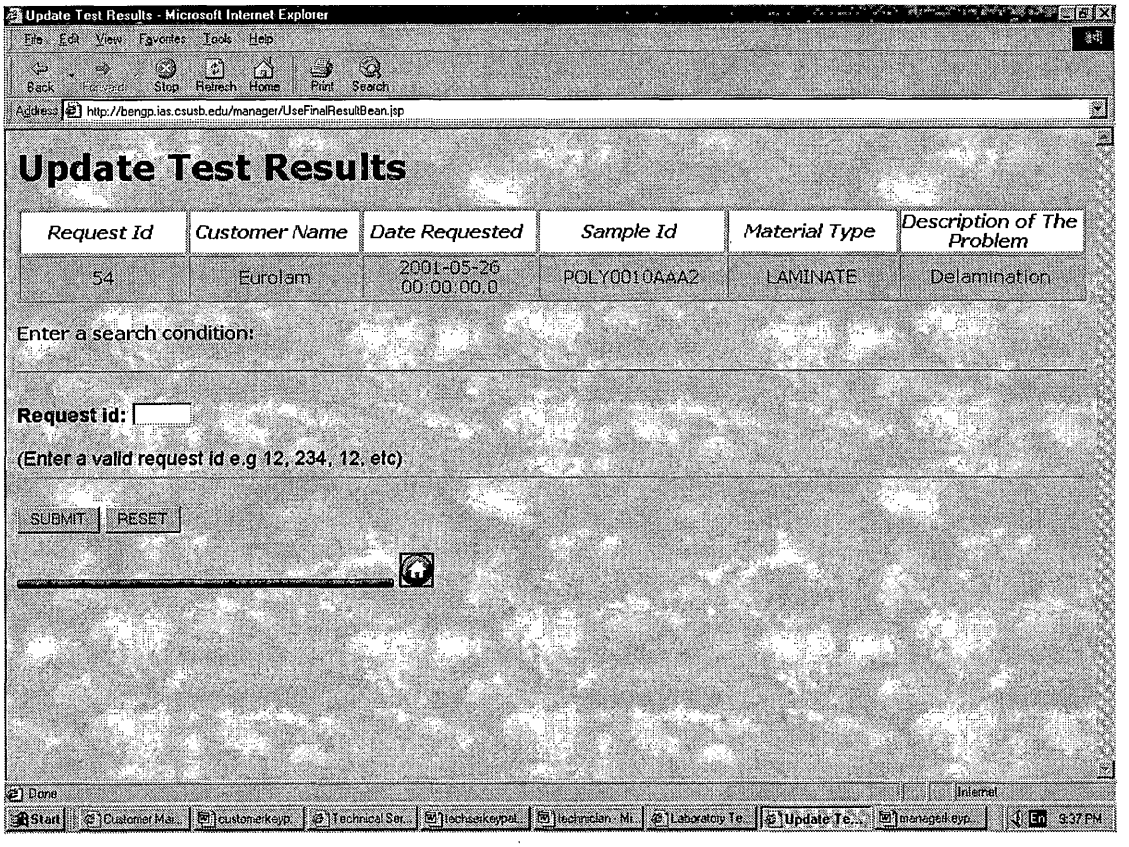
APPENDIX E

MANAGER KEY PATH

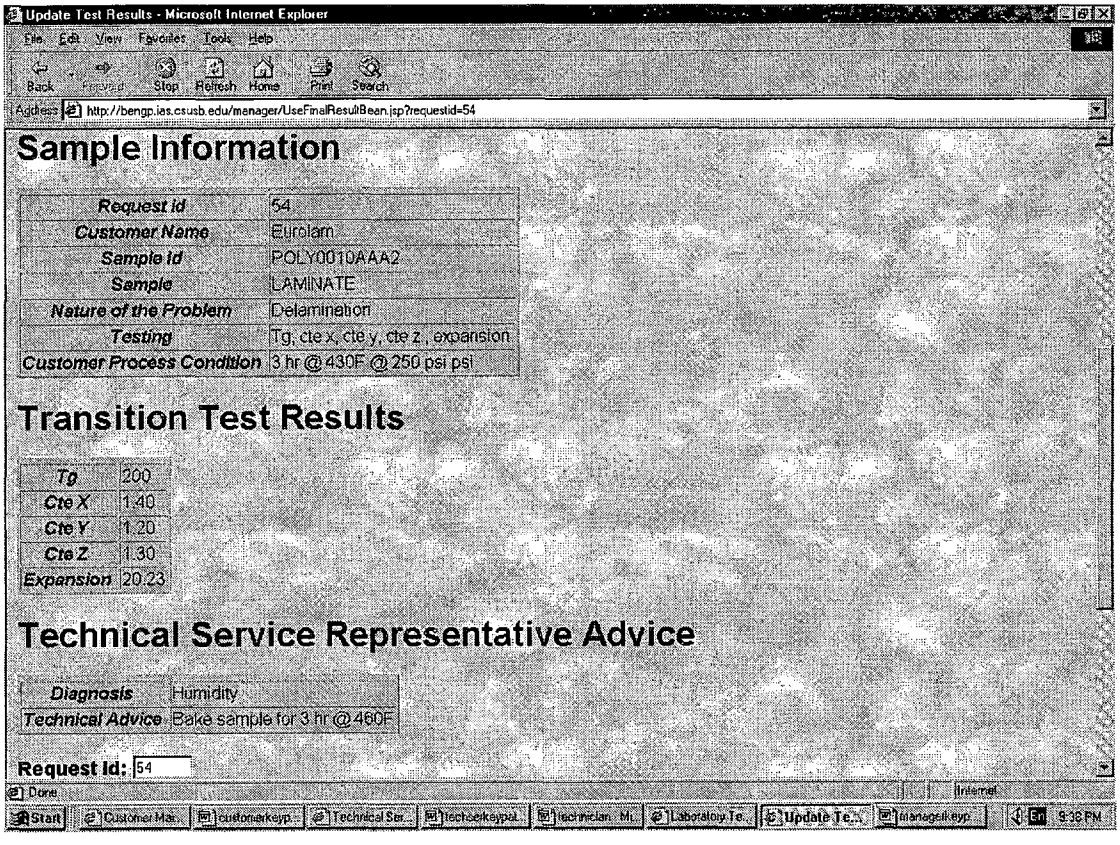
MANAGER MAIN MENU



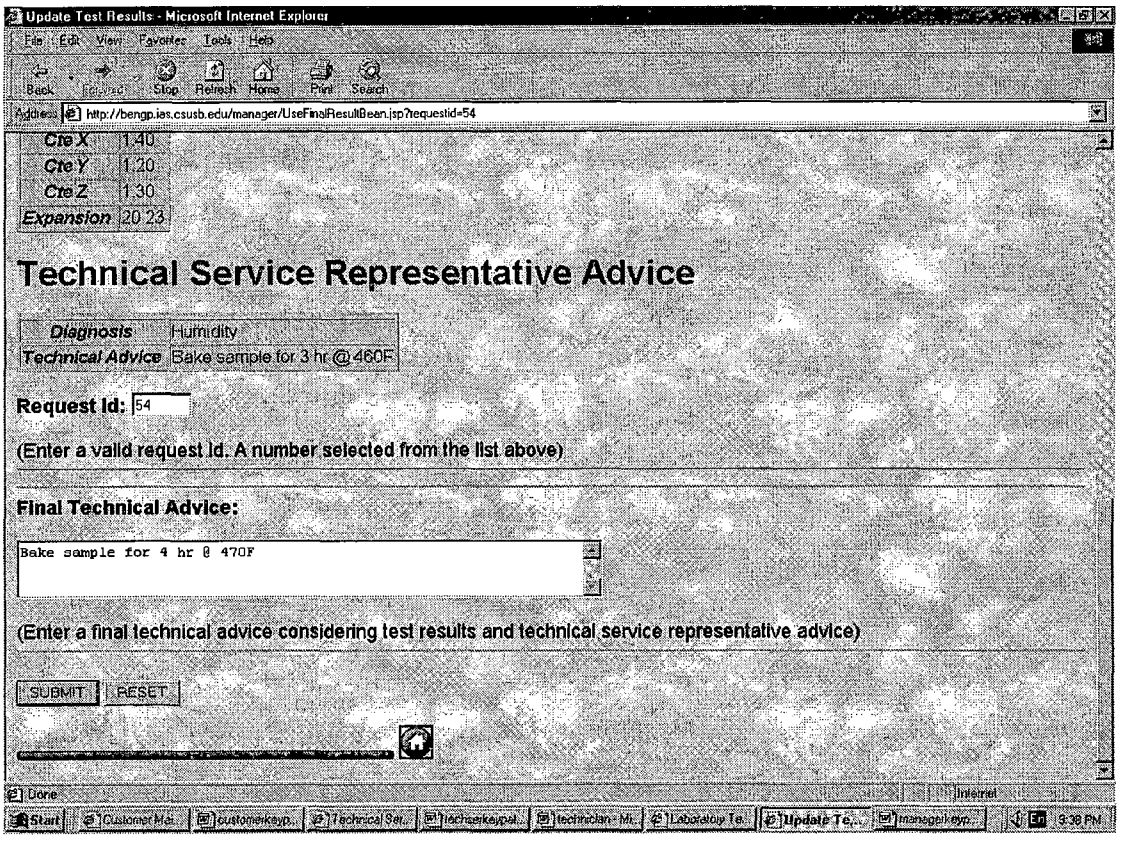
UPDATE TEST RESULTS



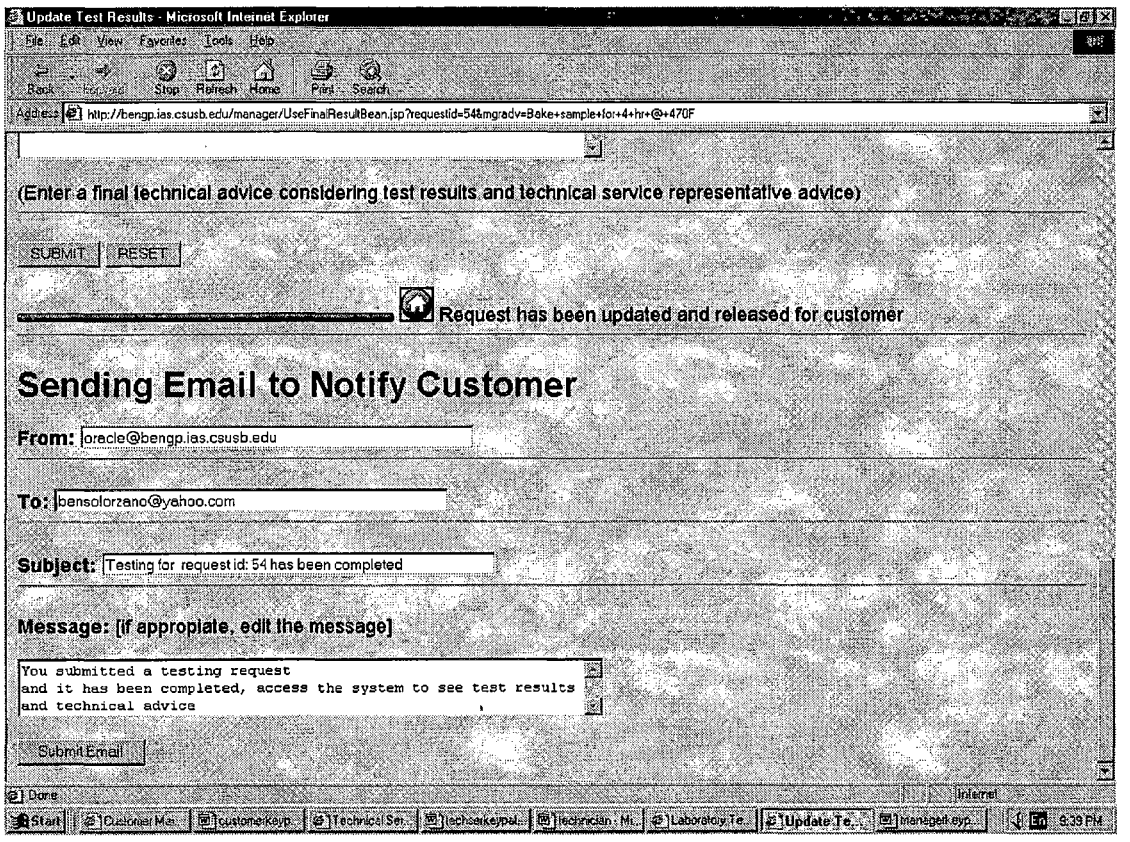
UPDATE TEST RESULTS



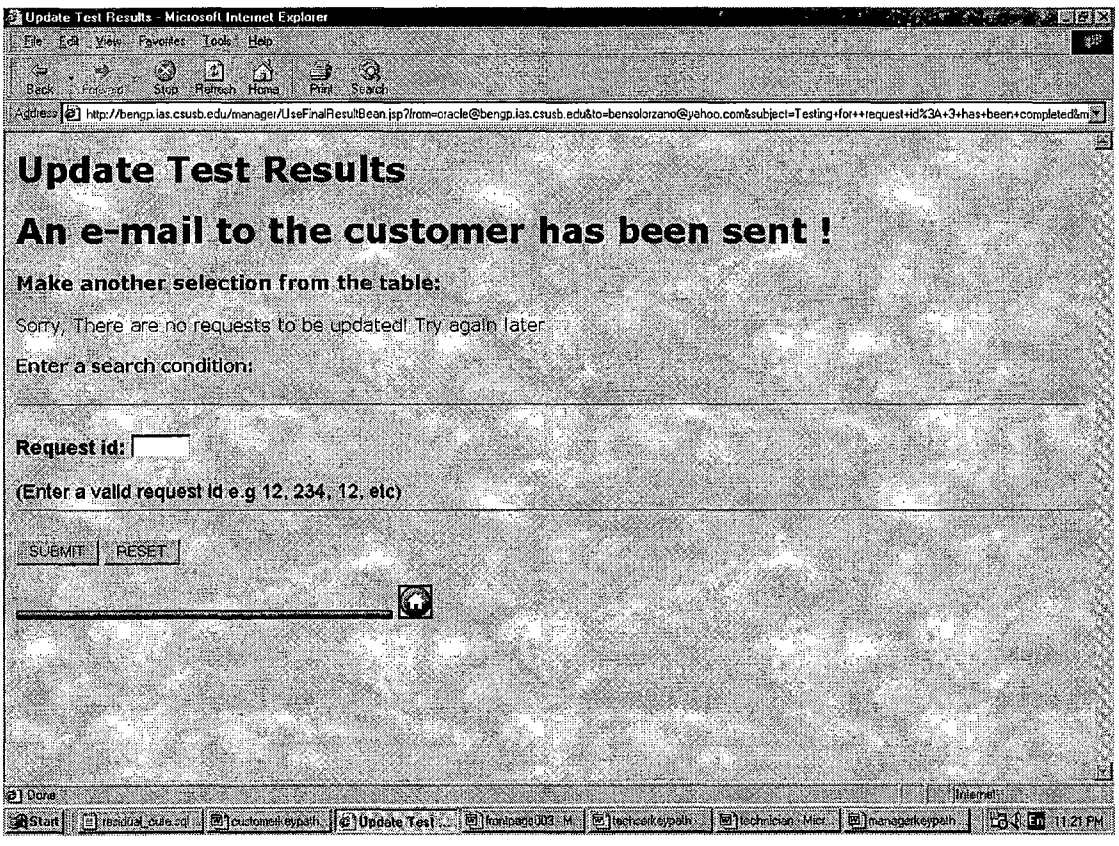
UPDATE TEST RESULTS



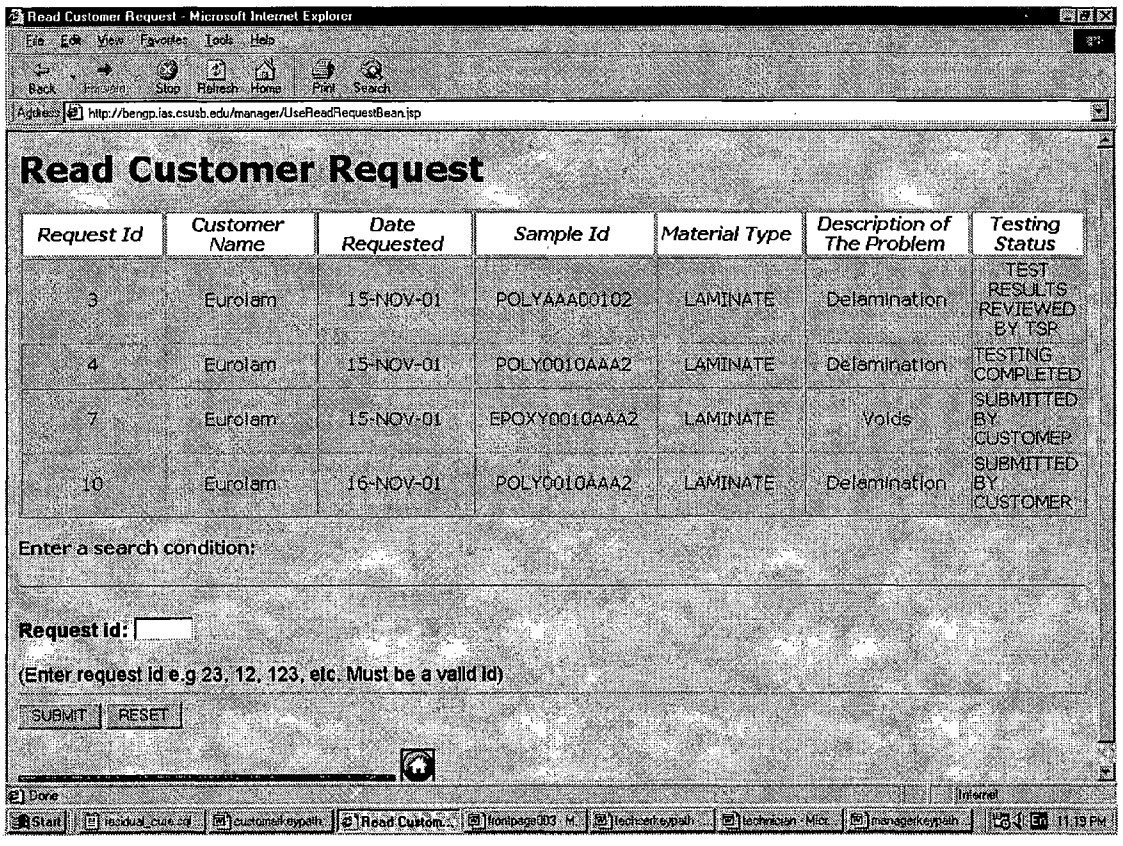
UPDATE TEST RESULTS



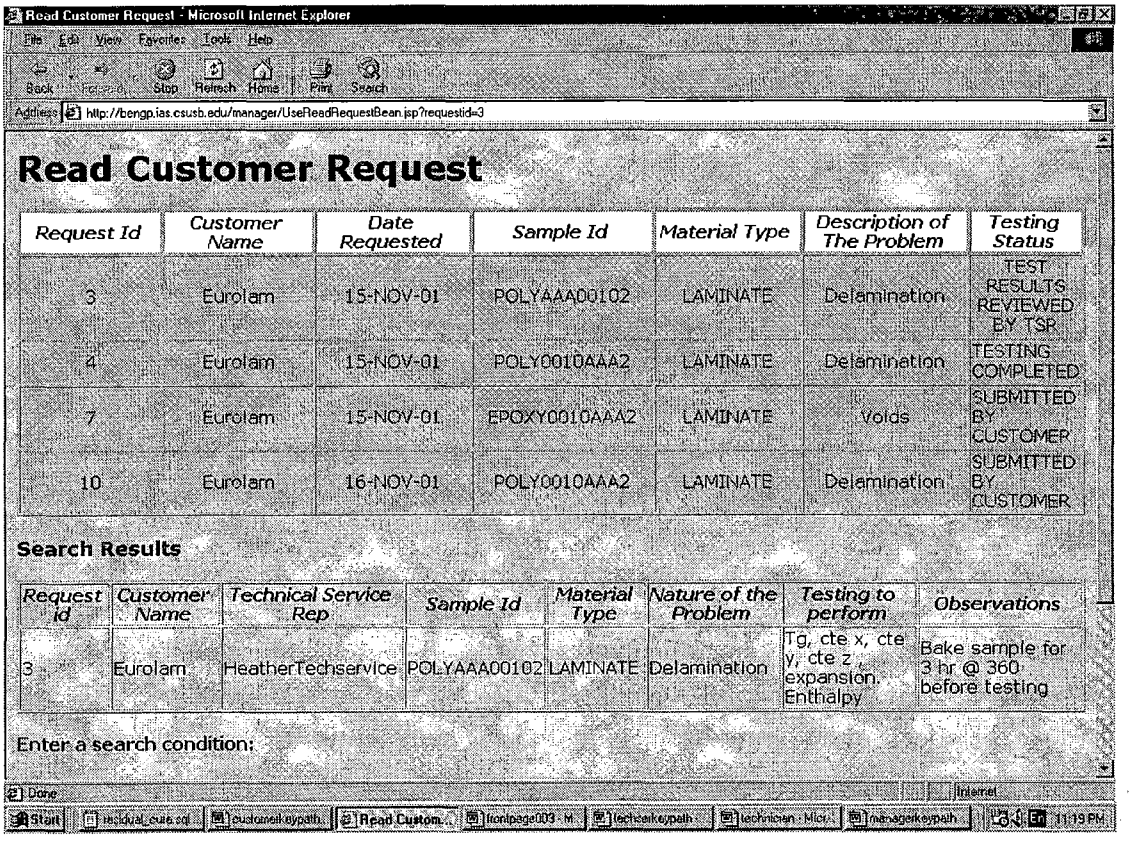
SENDING EMAIL



READ CUSTOMER REQUEST



READ CUSTOMER REQUEST



APPENDIX F

SOURCE CODE: JAVA SERVER PAGES FILES AND JAVA BEANS

key path: customer
file: index.jsp

```
<%@page import="java.sql.*" %>
<%
  String userid=request.getParameter("userid");
  String connStr=request.getParameter("connStr");
  if (connStr==null || userid==null) {
    connStr=(String)session.getValue("connStr");
    userid=(String)session.getValue("userid");
  }
  session.putValue("connStr", connStr);
  session.putValue("userid", userid);
  if (connStr==null || userid==null) { %>
<jsp:forward page="/index.jsp" />
<% } %>
 <H1> Customer <I> <%=userid %> </I> !</H1>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <meta name="GENERATOR" content="Mozilla/4.72 [en] (X11; I; SunOS 5.6 sun4u)
[Netscape]">
  <title>Customer Main Menu</title>
</head>
<body background="/images/crinklep.jpg">
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr VALIGN=TOP>
<td WIDTH="100%">
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td>
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td COLSPAN="3"><a NAME="general"></a><b><font face="Arial, Helvetica"><font
size=+5>Customer Main Menu
</font></font></b></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica"><font size=+2>Add a Request
</font></font></b>
<br><font size=+1>The customer can request a test
on samples</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/customer/addrequest.jsp">Take a look</a></font></td>
</tr>
```

```
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
<font size=+2>Test Results</font></font></b>
<br><font size=+1>
The customer can request test results in samples submitted
previously
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/customer/UseTestResultBean.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
<font size=+2>Historical Data</font></font></b>
<br><font size=+1>
The customer can request historical data of all
testing performed for them
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/customer/UseHistoricalBean.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
```

```
<font size=+2>Exit</font></font></b>
<br><font size=+1>
Go back to main menu
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/index.html">Exit</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

</table>
</td>
</tr>
</table>
</td>

<td WIDTH="15"></td>

<td WIDTH="1"></td>

<td WIDTH="15"></td>

<td WIDTH="100"> </td>
</tr>
</table>
</body>
</html>


Key-path:customer
file:addrequest.jsp

<%@ page import="java.sql.*" %>
<!--------------------------------------------------------------
 * This is a basic JavaServer Page that does an INSERT into request table
 *
 *
--------------------------------------------------------------!>
<%
  connStr="jdbc:oracle:thin:@localhost:1521:ora1";
  DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
  Connection conn = DriverManager.getConnection(connStr,
                                "ben28", "benito29");
  Statement stmt0 = conn.createStatement();
```

```
  Statement stmt = conn.createStatement ();
  Statement stmt1 = conn.createStatement ();
  Statement stmt2 = conn.createStatement ();
  String userid = request.getParameter("userid");
  String sampleid = request.getParameter("sampleid");
  String materialtype = request.getParameter("materialtype");
  String construction = request.getParameter("construction");
  String wor_num_aff= request.getParameter("wor_num_aff");
  String oth_lay_mat= request.getParameter("oth_lay_mat");
  String nat_oft_pro= request.getParameter("nat_oft_pro");
  String cus_pro_con= request.getParameter("cus_pro_con");
  String lab_sup_req= request.getParameter("lab_sup_req");
  String technicianid = request.getParameter("technicianid");
  String mfgdate = request.getParameter("mfgdate");
  String flag = request.getParameter("flag");
  if (materialtype == null)
  {
    materialtype=(String)session.getValue("materialtype");
  }
  else {
    session.putValue("materialtype",materialtype);
  } %>
  <%
  if (userid==null){
    userid=(String)session.getValue("userid");
  }
  else {
    session.putValue("userid",userid);
  }
   %>
<HTML>
 <HEAD>
  <TITLE>
  Adding a Request
  </TITLE>
 </HEAD>
<BODY background="/images/crinklep.jpg" >
<H1>
 <%= (request.getRemoteUser() != null? ", " + request.getRemoteUser() : "") %>
Adding a request  !
</H1>
<HR>
<%
   if ((sampleid != null) && (!sampleid.equals("")) &&
      (!wor_num_aff.equals("")) && (!oth_lay_mat.equals("")) &&
      (!nat_oft_pro.equals("")) && (!cus_pro_con.equals("")) &&
      (!lab_sup_req.equals("")) &&
      (!construction.equals("")) && (userid != null) && (oth_lay_mat != null) &&
      (nat_oft_pro != null) && (cus_pro_con != null) && (lab_sup_req != null) &&
      (mfgdate != null) && (materialtype != null) && (construction != null) &&
      (wor_num_aff != null)) {
```

```
    if(mfgdate.length() > 11)
    { %> <font color="red"><H1>Manufacturing date must follow the format DD-MM-
YYYY</H1></font> <%}%>
    <%
  try {
    String sql0= "SELECT techsrvrep_id FROM customer WHERE user_id='" +userid+ "'";
    ResultSet rset0 = stmt0.executeQuery(sql0);
    String techsvrep = null;
    if(rset0.next())
    techsvrep =rset0.getString(1);

    String sql = "INSERT INTO request(request_id, sample_id, construction, " +
            "workorder_numbers_affected, other_layered_materials,  mfg_date, " +
            "technician_id, nature_ofthe_problem, lab_support_requested, " +
            "customer_process_condition, material_type, customer_id, " +
            "techsrvrep_id, testing_status,date_requested)"  +
            " VALUES (request_id_seq.NEXTVAL,'" +sampleid+ "','"+construction+"','"
            +wor_num_aff+"','"+oth_lay_mat+"','"
            +mfgdate+ "','Dummy','"
            +nat_oft_pro+ "','" +lab_sup_req+ "','"
            +cus_pro_con+ "','" +materialtype+ "','" +userid+ "','" +techsvrep+
             "','SUBMITTED BY CUSTOMER',SYSDATE)";

   int rows  = stmt.executeUpdate(sql);
 %>
<%
    String sql1 = "SELECT request_id_seq.CURRVAL " +
             "FROM dual";

    String  requestid = " ";
    ResultSet rset1 = stmt1.executeQuery(sql1);
    if(rset1.next()){
    requestid =rset1.getString(1);
    }
    String sql2 = "SELECT sample_id, nature_ofthe_problem" +
             " FROM request" +
             " WHERE request_id="+requestid;
    ResultSet rset2 = stmt2.executeQuery(sql2);


   sql = "INSERT INTO final_request(request_id,propertesting_toperform,observations," +
            "techsrvrep_id)" +
            " VALUES (" +requestid+ ",' ', ' ','" +techsvrep+ "')";
    rows = stmt.executeUpdate(sql);
    stmt.close();
    String query = " ";
    rset0.close();
    stmt0.close();
 %>
     <HR> <P> <B> Your request has been submitted succesfully !
```

```
      <P><H1><font color="blue"><blink>Print out this screen, using the print option of your
browser</blink></font></H1>
      <P> Use this request id to retrieve test results in the future !
         <TABLE BORDER=1 BGCOLOR="C0C0C0">
         <TR><TH WIDTH=200 BGCOLOR="white"> <I>Request Id</I> </TH>
      <TD ALIGN=CENTER> <%=requestid%> </TD> </TR>
      <TR><TH WIDTH=200 BGCOLOR="white"> <I>Customer Name</I></TH>
       <TD ALIGN=CENTER> <%=userid %> </TD> </TR>
      <% if(rset2.next()){%>
      <TR><TH WIDTH=200 BGCOLOR="white"> <I>Sample Id</I> </TH>
      <TD ALIGN=CENTER> <%=rset2.getString(1)%> </TD></TR>
      <TR><TH WIDTH=200 BGCOLOR="white"> <I>Description of the problem</I></TH>
      <TD ALIGN=CENTER> <%=rset2.getString(2)%> </TD></TR> <% } %>
    </TABLE>
    <IMG SRC="/images/Normativ8D1.gif">
    <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<%
    rset1.close();
    stmt.close();
    stmt1.close();
    stmt2.close();
  } catch (SQLException e) {
    out.println("<P>" + "<font color=red><H1>The date format you entered is not valid" +
            " Enter the value again  !</H1></font>");
    out.println ("<font color=red><H1><PRE>" + e + "</PRE></H1></font> \n <P>");
%>
 <FORM METHOD=get>
   <font face="Arial, Helvetica"><font size=+1>

   <P><b>Customer Name:</b><INPUT TYPE="text" NAME="flag" SIZE=15
VALUE="<%=userid%>" >
   <HR>
    <P><b>Sample Identification:</b><INPUT TYPE="text" NAME="sampleid" SIZE=15
<%if(flag == null) { %>
    VALUE="<%%>"<%}%>
    <% if (flag != null) {%> VALUE="<%=sampleid%>" <% } %> >
    <% if (flag != null && sampleid.equals("")) { %><font color="red"><b><----  <blink>Must
Enter Sample Id</b></blink></font>
    <% } %>
   <P>(Label that includes characters and numbers which identifies the sample to be tested
      e.g. POLY0010AAA2
      POLY: for polyimide and EPOX for epoxy
      0010: Thickness (inches)
      AAA2 : Arbitrary string of 4 characters, given by the customer)
   <HR>
   <P><b>Select Material Type:</b> <% if(flag != null && materialtype == null) { %>
   <font color="red" ><b>  <--- <blink> Need to select Material Type</b></blink></font> <% }
%>
   <P><INPUT TYPE="radio" NAME="materialtype" VALUE="MULTILAYER-
BOARD">Multilayer board
   <INPUT TYPE="radio" NAME="materialtype" VALUE="LAMINATE">Laminate
```

```
   <HR>
   <P><b>Material Construction:</b> <INPUT TYPE="text" NAME="construction" SIZE=50 <%
if(flag == null) {%>
    VALUE="<%%>" <%}%>
    <%if (flag != null) {%> VALUE="<%=construction%>" <% } %> >
   <% if(flag != null && construction.equals("")) { %>
   <font color="red" ><b>  <---  <P><blink> Must Enter material
construction</b></blink></font> <% } %>
   <P>(Amount and type of laminates and prepregs which were used to manufacture the
product e.g
     1 ply of polyimide 1080 prepreg and 1 ply of epoxy 7628 prepreg)
   <HR>
   <P><b>Work Order Numbers Affected:</b> <INPUT TYPE="text" NAME="wor_num_aff"
SIZE=50 <% if(flag == null) {%>
   VALUE="<%%>" <%}%>
    <% if (flag != null) { %> VALUE="<%=wor_num_aff%>" <%}%>  >
   <%  if (flag != null && wor_num_aff.equals("")) { %>
                                                          <font color="red" ><b>
<--- <P><blink> Must Enter  Work Order Numbers Affected</b></blink></font>
    <% } %>
   <P>(Work order numbers of each of the elements(prepregs/laminates) of the product e.g.
     1080 polyimide-prepreg: 234354, 7628 epoxy-prepreg: 234321)
   <HR>
   <P><b>Enter other layered materials:</b>
   <INPUT TYPE="text"
        NAME="oth_lay_mat" SIZE=50 <% if(flag == null) {%>
   VALUE ="<%%>" <%}%>
   <% if (flag != null) { %> VALUE="<%=oth_lay_mat%>" <%}%>  >
<% if(flag != null && oth_lay_mat.equals("")) { %>
 <font color="red"><b> <--- <P><blink> Must Enter Other Layered
Materials</b></blink></font><% } %>

   <P>(Other type of material included in the multilayer board other than laminates or
prepregs)
   <HR>
   <P><b>Enter Manufacturing Date:</b><INPUT TYPE="text" NAME="mfgdate" SIZE=11
<%if(flag == null){ %>
   VALUE="<%%>" <%}%>
   <% if (flag != null) { %> VALUE="<%=mfgdate%>" <%}%> >
    <% if (flag != null && mfgdate.equals("")) { %>
    <font color="red" ><b> <--- <blink> Must enter a manufacturing date</b></blink> </font>
    <% } %>
   <P>(Must follow this format e.g 28-MAY-2001, 03-JUN-1999, 10-DEC-2001, etc)
   <HR>
   <HR><b>Describe the nature of the problem:</b>
    <% if (flag != null && nat_oft_pro.equals("")) { %>
    <font color="red" ><b><--- <blink> Must enter nature of the problem</b></blink> </font>
    <% } %>
   <P> <TEXTAREA NAME="nat_oft_pro" ROWS=4 COLS=65 <%if(flag==null){%>
WRAP></TEXTAREA><%}%>
   <% if(flag!=null){%> WRAP><%=nat_oft_pro%></TEXTAREA><%}%>
```

```
   <P>(Brief summary describing the nature of
     the manufacturing problem)
   <HR><b>Customer Process Conditions:</b> <% if (flag != null && cus_pro_con.equals("")) {
%>
   <font color="red"><b> <--- <blink>Must enter Customer Process
Conditions</b></blink></font>
   <% } %>
   <P><TEXTAREA NAME="cus_pro_con" ROWS=4
                                                          COLS=65
<%if(flag==null){%> WRAP></TEXTAREA><%} if(flag!=null){%>
WRAP><%=cus_pro_con%></TEXTAREA><%}%>
   <P>(Manufacturing conditions. It includes temperature, time and
     pressure e.g. 90 minutes @ 360C at 15 psi kiss pressure)
   <HR>
   <P><b>Lab Support Requested:</b> <INPUT TYPE="text" NAME="lab_sup_req"
SIZE=50<%if(flag==null){%>
   VALUE="<%%>" <%}%>
   <% if(flag!=null){%> VALUE="<%=lab_sup_req%>" <%}%>  >
   <% if(flag != null && lab_sup_req.equals("")) { %>
   <font color="red"><b><---  <p><blink> Must enter Lab Support Requested
</b></blink></font> <% } %>
   <P>(Type of testing which according to the customer will solve the problem)
   <HR>
   <P><INPUT TYPE="submit" VALUE="SUBMIT">
     <INPUT TYPE="reset" VALUE="RESET">
   </FORM>
   <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%
 }
   }
   else
   {
    if ( flag != null)
   {  %><font color="red"><blink> <H1>Entries marked with a red arrow cannot be empty
</H1></blink></font> <% } %>
   <FORM METHOD=get>
   <font face="Arial, Helvetica"><font size=+1>

   <P><b>Customer Name:</b><INPUT TYPE="text" NAME="flag" SIZE=15
VALUE="<%=userid%>" >
   <HR>
    <P><b>Sample Identification:</b><INPUT TYPE="text" NAME="sampleid" SIZE=15
<%if(flag == null) { %>
    VALUE="<%%>"<%}%>
    <% if (flag != null) {%> VALUE="<%=sampleid%>" <% } %> >
    <% if (flag != null && sampleid.equals("")) { %><font color="red"><b><----  <blink>Must
Enter Sample Id</b></blink></font>
    <% } %>
   <P>(Label that includes characters and numbers which identifies the sample to be tested
```

```
    e.g. POLY0010AAA2
    POLY: for polyimide and EPOX for epoxy
    0010: Thickness (inches)
    AAA2 : Arbitrary string of 4 characters, given by the customer)
  <HR>
  <P><b>Select Material Type:</b> <% if(flag != null && materialtype == null) { %>
  <font color="red" ><b>  <--- <blink> Need to select Material Type</b></blink></font> <% }
%>
  <P><INPUT TYPE="radio" NAME="materialtype" VALUE="MULTILAYER-
BOARD">Multilayer board
  <INPUT TYPE="radio" NAME="materialtype" VALUE="LAMINATE">Laminate
  <HR>
  <P><b>Material Construction:</b> <INPUT TYPE="text" NAME="construction" SIZE=50 <%
if(flag == null) {%>
   VALUE="<%%>" <%}%>
   <%if (flag != null) {%> VALUE="<%=construction%>" <% } %> >
  <% if(flag != null && construction.equals("")) { %>
  <font color="red" ><b>  <---  <P><blink> Must Enter material
construction</b></blink></font> <% } %>
  <P>(Amount and type of laminates and prepregs which were used to manufacture the
product e.g
    1 ply of polyimide 1080 prepreg and 1 ply of epoxy 7628 prepreg)
  <HR>
  <P><b>Work Order Numbers Affected:</b> <INPUT TYPE="text" NAME="wor_num_aff"
SIZE=50 <% if(flag == null) {%>
  VALUE="<%%>" <%}%>
   <% if (flag != null) { %> VALUE="<%=wor_num_aff%>" <%}%>  >
  <%  if (flag != null && wor_num_aff.equals("")) { %>
   <font color="red" ><b> <--- <P><blink> Must Enter  Work Order Numbers
Affected</b></blink></font>
   <% } %>
  <P>(Work order numbers of each of the elements(prepregs/laminates) of the product e.g.
    1080 polyimide-prepreg: 234354, 7628 epoxy-prepreg: 234321)
  <HR>
  <P><b>Enter other layered materials:</b>
  <INPUT TYPE="text"
       NAME="oth_lay_mat" SIZE=50 <% if(flag == null) {%>
  VALUE ="<%%>" <%}%>
  <% if (flag != null) { %> VALUE="<%=oth_lay_mat%>" <%}%>  >
<% if(flag != null && oth_lay_mat.equals("")) { %>
 <font color="red"><b> <--- <P><blink> Must Enter Other Layered
Materials</b></blink></font><% } %>

  <P>(Other type of material included in the multilayer board other than laminates or
prepregs)
  <HR>
  <P><b>Enter Manufacturing Date:</b><INPUT TYPE="text" NAME="mfgdate" SIZE=11
<%if(flag == null){ %>
  VALUE="<%%>" <%}%>
  <% if (flag != null) { %> VALUE="<%=mfgdate%>" <%}%> >
   <% if (flag != null && mfgdate.equals("")) { %>
```

```
    <font color="red" ><b> <--- <blink> Must enter a manufacturing date</b></blink> </font>
    <% } %>
   <P>(Must follow this format e.g 28-MAY-2001, 03-JUN-1999, 10-DEC-2001, etc)
   <HR>
   <HR><b>Describe the nature of the problem:</b>
    <% if (flag != null && nat_oft_pro.equals("")) { %>
    <font color="red" ><b><--- <blink> Must enter nature of the problem</b></blink> </font>
    <% } %>
   <P> <TEXTAREA NAME="nat_oft_pro" ROWS=4 COLS=65 <%if(flag==null){%>
WRAP></TEXTAREA><%}%>
   <% if(flag!=null){%> WRAP><%=nat_oft_pro%></TEXTAREA><%}%>
   <P>(Brief summary describing the nature of
     the manufacturing problem)
   <HR><b>Customer Process Conditions:</b> <% if (flag != null && cus_pro_con.equals("")) {
%>
   <font color="red"><b> <--- <blink>Must enter Customer Process
Conditions</b></blink></font>
   <% } %>
   <P><TEXTAREA NAME="cus_pro_con" ROWS=4
      COLS=65 <%if(flag==null){%> WRAP></TEXTAREA><%} if(flag!=null){%>
WRAP><%=cus_pro_con%></TEXTAREA><%}%>
   <P>(Manufacturing conditions. It includes temperature, time and
     pressure e.g. 90 minutes @ 360C at 15 psi kiss pressure)
   <HR>
   <P><b>Lab Support Requested:</b> <INPUT TYPE="text" NAME="lab_sup_req"
SIZE=50<%if(flag==null){%>
   VALUE="<%%>" <%}%>
   <% if(flag!=null){%> VALUE="<%=lab_sup_req%>" <%}%>  >
   <% if(flag != null && lab_sup_req.equals("")) { %>
   <font color="red"><b><---  <p><blink> Must enter Lab Support Requested
</b></blink></font> <% } %>
   <P>(Type of testing which according to the customer will solve the problem)
   <HR>
   <P><INPUT TYPE="submit" VALUE="SUBMIT">
     <INPUT TYPE="reset" VALUE="RESET">
   </FORM>
   <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<%
   }
%>

 </BODY>
</HTML>
Key-path:customer
file:UseTestResultBean.jsp
<%@ page import="java.sql.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String connStr=request.getParameter("connStr");
```

```
      connStr="jdbc:oracle:thin:@localhost:1521:ora1";
   DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
   Connection conn = DriverManager.getConnection(connStr,
                                        "ben28", "benito29");
    Statement stmt = conn.createStatement();
    if(connStr==null) {
    connStr=(String)session.getValue("connStr");
   } else {
    session.putValue("connStr",connStr);
   }
   if (connStr==null) { %>
<jsp:forward page="/index.jsp"/>
<%
   } %>
<jsp:useBean id="TestResultBean" class="beans.TestResultBean" scope="session" />
<jsp:setProperty name="TestResultBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="TestResultBean" property="userid"  />
<jsp:setProperty name="TestResultBean" property="requestid"  />
<jsp:setProperty name="TestResultBean" property="customerid" />

<HTML>
  <HEAD>
    <TITLE>
       Update Test Results
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1>  Test Results </H1>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<HR>
<P><b>Request id:</b> <INPUT TYPE=text NAME="requestid" SIZE=6 VALUE="<%%>" >
<P>(Enter a valid request id e.g 12, 234, 12, etc)
<HR>
<P><INPUT TYPE=submit VALUE="SUBMIT">
<INPUT TYPE=reset VALUE="RESET">
</FORM>
 <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%
   if (requestid != null) {
  String tg = null;
  String enthalpyCheck = null;
  String t300min = null;
  String t288min = null;
  String t260min = null;
    try {

    String query = "SELECT tr.tg FROM request r, final_request fr, historical h, " +
                " testing t, transition tr " +
```

```
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=tr.testing_id " +
            " AND t.test_type='TRANSITION'" +
            " AND t.historical_id='" +requestid+ "'";
  ResultSet  rset = stmt.executeQuery(query);
  if(rset.next())
  tg = rset.getString(1);
      query = "SELECT rc.enthalpy FROM request r, final_request fr, historical h, " +
          " testing t, residual_cure rc " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=rc.testing_id " +
            " AND t.test_type='RESIDUAL'" +
            " AND t.historical_id='" +requestid+ "'";
         rset = stmt.executeQuery(query);
  if(rset.next())
  enthalpyCheck = rset.getString(1);
query = "SELECT res.t300_min, res.t288_min, res.t260_min " +
      " FROM request r, final_request fr, historical h, "+
          " testing t, resistance res " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=res.testing_id " +
            " AND t.test_type='RESISTANCE'" +
            " AND t.historical_id='" +requestid+ "'";

         rset = stmt.executeQuery(query);
  if(rset.next())
  {
  t300min = rset.getString(1);
  t288min = rset.getString(2);
  t260min = rset.getString(3);
  }
  stmt.close();
  rset.close();
     }

   catch (SQLException e) {
       out.println("<P>" + "This was an error doing the query:");
      out.println ("<PRE>" + e + "</PRE> \n <P>");
    } %>

  <H1>Sample Information</H1>
  <%= TestResultBean.getResultInfo() %>
<% if(tg != null) { %>
  <H1>Transition Test Results</H1>
  <%= TestResultBean.getResultTransition() %>
```

```
 <% }  if (t300min != null) { %>

    <H1>T300 Test Results</H1>
    <%= TestResultBean.getResultT300() %>
 <% } if (t288min != null) { %>
    <H1> T288 Test Results </H1>
    <%= TestResultBean.getResultT288() %>
 <% } if (t260min != null) { %>
    <H1> T260 Test Results </H1>
    <%= TestResultBean.getResultT260() %>
 <% } if (enthalpyCheck != null) { %>
    <H1>Residual Cure Test Results</H1>
    <%= TestResultBean.getResultResidual() %>
 <% }  %>
    <H1> Technical Advice </H1>
    <%= TestResultBean.getResultAdvice() %>

<% } %>
</BODY>
</HTML>
```

Keypath:customer
file:UseHistoricalBean.jsp

```
<%@ page import="java.sql.*" %>
<%
 String connStr=request.getParameter("connStr");
 connStr="jdbc:oracle:thin:@localhost:1521:ora1";   DriverManager.registerDriver(new
oracle.jdbc.driver.OracleDriver());
  Connection conn = DriverManager.getConnection(connStr,
                                    "ben28", "benito29");
  String material=request.getParameter("material");
  String date=request.getParameter("date");
  String test=request.getParameter("test");
  String statics=request.getParameter("statics");
  String userid=request.getParameter("userid");
  String typeoftest=request.getParameter("typeoftest");
  String flag = request.getParameter("flag");
  String t300 = "t300";
  if (userid==null){
    userid=(String)session.getValue("userid");
   }else {
     session.putValue("userid",userid);
   }
  if (userid==null) { %>
  <jsp:forward page="/index.jsp" />
  <% } %>
<jsp:useBean id="historicalBean" class="beans.HistoricalBean" scope="session" />
<jsp:setProperty name="historicalBean" property="date" value="<%=date%>" />
<jsp:setProperty name="historicalBean" property="userid" value="<%=userid%>" />
<jsp:setProperty name="historicalBean" property="material" value="<%=material%>"  />
<jsp:setProperty name="historicalBean" property="test" value="<%=test%>" />
```

```
<jsp:setProperty name="historicalBean" property="statics" value="<%=statics%>"  />
<jsp:setProperty name="historicalBean" property="connStr" value="<%=connStr%>"  />
<jsp:setProperty name="historicalBean" property="typeoftest" value="<%=typeoftest%>" />
  <HTML>
  <HEAD><TITLE> Customer </TITLE> </HEAD>
  <BODY background="/images/sky.jpg">
<%
   if (date != null && material != null && test != null && statics != null ) {

      session.putValue("connStr",connStr);
      session.putValue("userid",userid);
      %>
      <H3> CUSTOMER HISTORICAL DATA </H3>
      <H3>TEST RESULTS FOR: <I><%=userid%> </I></H3>
      <H3>TEST TYPE: <I><%=test%> </I></H3>
      <H3>DATE: <I><%=date%> </I></H3>
      <%
      String staticsyes = "YES";
      String staticsno = "NO";
       if(statics.compareTo(staticsno) == 0) {
      %>
      <%= historicalBean.getResult()%>
       <% }
       else { %>
      <%= historicalBean.getResult()%>
      <%= historicalBean.getStatistics()%>
      <% } %>
      <HR>
      <IMG SRC="/images/Normativ8D1.gif">
      <A HREF="UseHistoricalBean.jsp"><IMG SRC="/images/BlurMet7D4.gif"></A>
      <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%  } else {
%>
<HTML>
<HEAD> <TITLE> Customer Historical Data </TITLE> </HEAD>
<H1><b> Customer Historical Data </b> </H1>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<INPUT TYPE="TEXT" NAME="flag" SIZE=25 VALUE="Historical Data For <%=userid%>"
><HR>
<H3><B>Enter a search condition:</B></H3>
<P><b>Select Date :</b> <INPUT TYPE="TEXT" NAME="date" SIZE=11 <%if(date == null ||
date.equals("")){%> VALUE="<%%>" <%}%>
<%if(flag != null && !date.equals("")){ %> VALUE="<%=date%>" <%}%> >
<%
  if (flag != null && date.equals("")){
%> <font color=red><blink><--- Must enter a date</blink></font> <%}%>
<P>(Format: DD,MMM,YYYY e.g 01-JAN-2000, 23,SEP-2001, etc. The query will retrieve data
from this date up to now)
<HR><b>Select material type:  </b>
```

```
<INPUT TYPE="radio" NAME="material" VALUE="LAMINATE">LAMINATE
<INPUT TYPE="radio" NAME="material" VALUE="MULTILAYER-BOARD">MULTILAYER-
BOARD
<%
  if(material == null && date != null) {
%> <font color = red><blink> <--- Select Material Type </blink> </font> <%}%>
<HR><b>Select type of test:  </b>
<INPUT TYPE="radio" NAME="test" VALUE="TRANSITION">TRANSITION
<INPUT TYPE="radio" NAME="test" VALUE="RESIDUAL">RESIDUAL
<P><INPUT TYPE="radio" NAME="test" VALUE="RESISTANCE">RESISTANCE
Test type: <INPUT TYPE="TEXT" NAME="typeoftest" SIZE=4 VALUE="<%=t300%>">
<P>(If Resistance is selected, enter the type of test: t300 or t288 or t260)
<%
  if(test == null && flag != null) {
%> <font color = red><blink> <--- Select Test Type </blink> </font> <%}%><HR>
<P><b>Statistical Summary: </b>
<INPUT TYPE="radio" NAME="statics" VALUE="YES">YES
<INPUT TYPE="radio" NAME="statics" VALUE="NO">NO
<%
  if(statics == null && flag != null) {
%> <font color = red><blink> <--- Select "Yes" or "No" for Statistical Summary  </blink> </font>
<%}%>


<P><HR>
<INPUT TYPE="submit" VALUE="SUBMIT">
<INPUT TYPE="reset" VALUE="RESET">
</FORM>

<IMG SRC="/images/Normativ8D1.gif">
<A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<% } %>
</BODY>
</HTML>
```

```
key-path:technician
file:index.jsp
<%@page import="java.sql.*" %>
<%
  String userid=request.getParameter("userid");
  String connStr=request.getParameter("connStr");
  if (connStr==null || userid==null) {
    connStr=(String)session.getValue("connStr");
```

```
    userid=(String)session.getValue("userid");
  }
  session.putValue("connStr", connStr);
  session.putValue("userid", userid);
  if (connStr==null || userid==null) { %>
<jsp:forward page="/index.jsp" />
<% } %>
<H1> Laboratory Technician <I> <%=userid %> </I></H1>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <meta name="GENERATOR" content="Mozilla/4.72 [en] (X11; I; SunOS 5.6 sun4u)
[Netscape]">
  <title>Laboratory Technician </title>
</head>
<body background="/images/sky.jpg">
 
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr VALIGN=TOP>
<td WIDTH="100%">
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td>

<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td COLSPAN="3"><a NAME="general"></a><b><font face="Arial, Helvetica"><font
size=+5>Laboratory Technician  Main Menu
</font></font></b></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica"><font size=+2>Read Customer Request
</font></font></b>
<br><font size=+1>Laboratory Technician  reads customer request to perform testing
When you enter this section if any request is available, it will be automatically  assigned for you
to
be tested !
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/technician/UseReadRequestBean.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
```

```
</tr>

<tr>
<td WIDTH="35"></td>


Key-path:technician
file:ReadRequestBean.jsp

<%@ page import="java.sql.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String connStr=request.getParameter("connStr");
    if(connStr==null) {
    connStr=(String)session.getValue("connStr");
    userid=(String)session.getValue("userid");
  } else {
    session.putValue("connStr",connStr);
    session.putValue("userid",userid);
  }
  if (connStr==null || userid==null) { %>
<jsp:forward page="/index.jsp" />
<%
  } %>
<jsp:useBean id="readRequestBean" class="beans.ReadRequestBean" scope="session" />
<jsp:setProperty name="readRequestBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="readRequestBean" property="userid"  />
<jsp:setProperty name="readRequestBean" property="requestid"  />
<HTML>
  <HEAD>
    <TITLE>
       Read Customer Request
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1> Read Customer Request </H1>
<%
 Connection conn = DriverManager.getConnection(connStr,"ben28","benito29");
 Statement stmt = conn.createStatement();
 Statement stmt1 = conn.createStatement();
 try{
    String sql= "SELECT r.request_id, r.customer_id, " +
              "r.sample_id,TO_CHAR(r.mfg_date), " +
              " r.material_type, r.customer_process_condition, " +
              " fr.propertesting_toperform, fr.observations " +
    "FROM request r, final_request fr, historical h " +
    " WHERE r.request_id=fr.request_id AND fr.request_id=h.historical_id" +
    " AND h.technical_advice = ' '  AND h.diagnosis = ' '" +
    " AND r.technician_id = 'Dummy' " +
    " ORDER BY r.request_id " ;
```

```
    ResultSet rset = stmt.executeQuery(sql);
    if(rset.next()){
    String sql1 = "UPDATE  request" +
              " SET (technician_id)='"+userid+"'"+
              " WHERE request_id="+rset.getString(1);
    String sql2 = "UPDATE request" +
              " SET (testing_status)='TESTING IN PROGRESS'" +
              " WHERE request_id="+rset.getString(1);
    stmt1.executeUpdate(sql1);
    stmt1.executeUpdate(sql2);
%>
   <P><b> This request has been assigned for you to be tested </b>
   <TABLE BORDER=1 BGCOLOR="C0C0C0">
   <TH WIDTH=200 BGCOLOR="white"> <I> Request Id </I> </TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Customer Name </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Sample Id</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Manufacturing Date </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Material Type</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Customer Process Conditions</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Proper Testing to be performed</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Observations</I></TH>
   <TR> <TD ALIGN=CENTER> <%= rset.getString(1)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(2)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(3)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(4)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(5)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(6)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(7)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(8)%></TD>
   </TR>
    </TABLE>
<% }
   else {
%>
  <P> All requests have been assigned for now.
     Try again later ! </P>
<%
   }
   rset.close();

   stmt.close();
   stmt1.close();
  } catch (SQLException e) {
    out.println("<P>" + "There was an error doing the query:");
    out.println ("<PRE>" + e + "</PRE> \n <P>");
    }
%>

<%
   if (requestid != null) { %>
```

```
    <H3>Search Results</H3>
    <%=  readRequestBean.getResult() %>

<% } %>

<P><B><font color="blue"><blink> Remember !
Whenever a request is assigned for testing,  print the request using the
"print option" of your browser!
</blink></font>
</B></P>

 <IMG SRC="/images/Normativ8D1.gif">
    <A HREF="/technician/index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

</BODY>
</HTML>

key-path:technician
file: UseReportTestBean.jsp
<%@ page import="java.sql.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String propertest=request.getParameter("propertest");
    String observations=request.getParameter("observations");
    String connStr=request.getParameter("connStr");
    String tg=request.getParameter("tg");
    String ctex=request.getParameter("ctex");
    String ctey=request.getParameter("ctey");
    String ctez=request.getParameter("ctez");
    String expansion=request.getParameter("expansion");
    String t300=request.getParameter("t300");
    String t288=request.getParameter("t288");
    String t260=request.getParameter("t260");
    String t300min=request.getParameter("t300min");
    String t288min=request.getParameter("t288min");
    String t260min=request.getParameter("t260min");
    String enthalpy=request.getParameter("enthalpy");
    String enthalpyCheck= null;
    int value = 0;

    if(connStr==null  ) {
    connStr=(String)session.getValue("connStr");
    userid=(String)session.getValue("userid");
  } else {
    session.putValue("connStr",connStr);
    session.putValue("userid",userid);
  }
  if (connStr==null ) { %>
<jsp:forward page="/index.jsp" />
<%
```

```
  } %>
<jsp:useBean id="reportTestBean" class="beans.ReportTestBean" scope="session" />
<jsp:setProperty name="reportTestBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="reportTestBean" property="userid"  />
<jsp:setProperty name="reportTestBean" property="requestid"  />
<jsp:useBean id="updateTestBean" class="beans.ReportTestBean" scope="session" />
<jsp:setProperty name="updateTestBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="updateTestBean" property="userid" />
<jsp:setProperty name="updateTestBean" property="tg" />
<jsp:setProperty name="updateTestBean" property="requestid" />
<jsp:setProperty name="updateTestBean" property="ctex" />
<jsp:setProperty name="updateTestBean" property="ctey" />
<jsp:setProperty name="updateTestBean" property="ctez" />
<jsp:setProperty name="updateTestBean" property="expansion" />
<jsp:setProperty name="updateTestBean" property="t300" />
<jsp:setProperty name="updateTestBean" property="t288" />
<jsp:setProperty name="updateTestBean" property="t260" />
<jsp:setProperty name="updateTestBean" property="t300min" />
<jsp:setProperty name="updateTestBean" property="t288min"  />
<jsp:setProperty name="updateTestBean" property="t260min"  />
<jsp:setProperty name="updateTestBean" property="enthalpy" />
<jsp:setProperty name="updateTestBean" property="userid" />
<jsp:useBean id="recordBean" class="beans.ReportTestBean" scope="session" />
<jsp:setProperty name="recordBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="recordBean" property="requestid" />
<jsp:setProperty name="recordBean" property="userid" />
<HTML>
  <HEAD>
    <TITLE>
       Read Customer Request
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1> Reporting Customer Test Results </H1>
<%
 Connection conn = DriverManager.getConnection(connStr,"ben28","benito29");
 Statement stmt = conn.createStatement();
 if (expansion != null || t300 != null || t288 != null || t260 != null || enthalpy != null) { %>
    <%=updateTestBean.getUpdate()%>
 <H3> Make another selection: </H3>
<% }
 try{
   String sql= "SELECT r.request_id, r.customer_id,TO_CHAR(r.date_requested)," +
               "r.sample_id, r.material_type, r.nature_ofthe_problem" +
               " FROM request r " +
               " WHERE r.technician_id='"+userid+"'" +
               " AND r.testing_status='TESTING IN PROGRESS'"+
               " ORDER BY r.date_requested" ;
   ResultSet rset = stmt.executeQuery(sql);
   if(rset.next()){
%>
```

```
   <TABLE BORDER=1 BGCOLOR="C0C0C0">
   <TH WIDTH=200 BGCOLOR="white"> <I> Request Id </I> </TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Customer Name </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Date Requested</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Sample Id </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Material Type</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Description of The Problem</I></TH>
   <TR> <TD ALIGN=CENTER> <%= rset.getString(1)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(2)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(3)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(4)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(5)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(6)%></TD>
   </TR>
<%   while (rset.next()) {
%>
    <TR> <TD ALIGN=CENTER> <%= rset.getString(1) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(2) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(3) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(4) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(5) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(6) %></TD>
     </TR>
<% }
%>
    </TABLE>
<%  }
   else {
%>
  <P> There are no requests to report test results for! </P>
<%
   }
   rset.close();
   stmt.close();
  } catch (SQLException e) {
    out.println("<P>" + "There was an error doing the query:");
    out.println ("<PRE>" + e + "</PRE> \n <P>");
    }
    finally { %>
<P><font face="Arial, Helvetica"><font size=+1><B>Enter request id to report test
results:</B></P>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<P><b>Request id:</b><INPUT TYPE=text NAME="requestid" SIZE=6 VALUE="<%%>" >
<HR>
<P><INPUT TYPE=submit VALUE="SUBMIT">
<INPUT TYPE=reset VALUE="RESET">
</FORM>
<IMG SRC="/images/Normativ8D1.gif">
<A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<%   } %>
```

```
<%
  if (requestid != null) {
 conn = DriverManager.getConnection(connStr,"ben28","benito29");
 stmt = conn.createStatement();
  try {

  String query = "SELECT tr.tg FROM request r, final_request fr, historical h, " +
            " testing t, transition tr " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=tr.testing_id " +
            " AND t.test_type='TRANSITION'" +
            " AND t.historical_id='" +requestid+ "'";
  ResultSet  rset = stmt.executeQuery(query);
  if(rset.next())
  tg = rset.getString(1);
      query = "SELECT rc.enthalpy FROM request r, final_request fr, historical h, " +
            " testing t, residual_cure rc " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=rc.testing_id " +
            " AND t.test_type='RESIDUAL'" +
            " AND t.historical_id='" +requestid+ "'";
          rset = stmt.executeQuery(query);
  if(rset.next())
  enthalpyCheck = rset.getString(1);

 query = "SELECT res.t300_min, res.t288_min, res.t260_min " +
       " FROM request r, final_request fr, historical h, "+
            " testing t, resistance res " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=res.testing_id " +
            " AND t.test_type='RESISTANCE'" +
            " AND t.historical_id='" +requestid+ "'";

          rset = stmt.executeQuery(query);
  if(rset.next())
  {
  t300min = rset.getString(1);
  t288min = rset.getString(2);
  t260min = rset.getString(3);
  }
  stmt.close();
  rset.close();
     }
```

```
    catch (SQLException e) {
        out.println("<P>" + "This was an error doing the query:");
       out.println ("<PRE>" + e + "</PRE> \n <P>");
     }
       %>
   <H1>ENTER TEST RESULTS</H1>
   <%= reportTestBean.getResult() %>
   <FORM METHOD=get>
   <font face="Arial, Helvetica"><font size=+1>
   <HR>
   <P><b>Request Id: </b> <INPUT TYPE=text NAME="requestid" SIZE=6
VALUE="<%=requestid%>" >
   <HR>
   <%
     if(tg != null)
   { %>
   <P><b>Tg [Celcius]:</b> <INPUT TYPE=text NAME="tg" SIZE=6 VALUE="<%=value%>"
>
   <P>(Enter the number in the following format e.g: 200.23, 128.00, 134.34, 290.23, etc)
   <HR>
   <P><b>cte_x [um/mC]:</b> <INPUT TYPE=text NAME="ctex" SIZE=6
VALUE="<%=value%>" >
   <P>(Enter the number in the following format: 3.12, 12.23, 7.08, 3.45, etc)
   <HR>
   <P><b>cte_y [um/mC]:</b> <INPUT TYPE=text NAME="ctey" SIZE=6
VALUE="<%=value%>" >
   <P>(Enter the number in the following format: 3.15, 2.34, 12.23, 9.23, etc)
   <HR>
   <P><b>cte_z [um/mC]:</b> <INPUT TYPE=text NAME="ctez" SIZE=6
VALUE="<%=value%>" >
   <P>(Enter the number in the folling format: 2.32, 23.45, 12.05, etc)
   <HR>
   <P><b>Expansion [%]:</b> <INPUT TYPE=text NAME="expansion" SIZE=5
VALUE="<%=value%>" >
   <P>(Enter the percentage with the following format: 12.23, 2.23, 34.43, etc)
   <% } %>
   <HR>
   <% if (t300min != null) { %>
   <P><b>T300 [minutes]:</b><INPUT TYPE="text" NAME="t300min" SIZE=2
VALUE="<%=value%>" >
   <INPUT TYPE="radio" NAME="t300" SIZE=4 VALUE="PASS">Pass
   <INPUT TYPE="radio" NAME="t300" SIZE=4 VALUE="FAIL">Fail
   <P>(Enter the number of minutes, using the following number format: 00.00 e.g 23.32,
12.02, 2.32, etc)
   <HR>
    <% }
     if (t288min != null) { %>
   <P><b>T288 [minutes]:</b><INPUT TYPE="text" NAME="t288min" SIZE=2
VALUE="<%=value%>" >
   <INPUT TYPE="radio" NAME="t288" SIZE=4 VALUE="PASS">Pass
   <INPUT TYPE="radio" NAME="t288" SIZE=4 VALUE="FAIL">Fail
```

```
    <P>(Enter the number of minutes, using the following format: 00.00 e.g 2.32, 12.21, 12.24,
etc)
    <HR>
     <% }
       if (t260min != null) {  %>
    <P><b>T260 [minutes]:</b><INPUT TYPE="text" NAME="t260min" SIZE=2
VALUE="<%=value%> " >
    <INPUT TYPE="radio" NAME="t260" SIZE=4 VALUE="PASS">Pass
    <INPUT TYPE="radio" NAME="t260" SIZE=4 VALUE="FAIL">Fail
    <P>(Enter the number of minutes, using the following format: 00.00 e.g 2.32, 23.23, 23.12,
etc)
    <%  }  if (enthalpyCheck != null) { %>
    <HR>
    <P><b>Enthalpy:</b><INPUT TYPE="text" NAME="enthalpy" SIZE=6
VALUE="<%=value%>" >
    <P>(Enter the number with the following format: 127.43, 2.23, 123.23, etc)
    <%
       }
     %>
    <HR>
    <INPUT TYPE=submit VALUE="SUBMIT">
    <INPUT TYPE=reset VALUE="RESET">
    </FORM>
    <IMG SRC="/images/Normativ8D1.gif">
    <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%

 }
 %>
</BODY>
</HTML>
Key-path: technician
file: UseInternalHistoricalBean.jsp
<%@ page import="java.sql.*" %>
<%
  String connStr=request.getParameter("connStr");
  connStr="jdbc:oracle:thin:@localhost:1521:ora1";
  DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
  Connection conn = DriverManager.getConnection(connStr,
                                    "ben28", "benito29");
  String material=request.getParameter("material");
  String date=request.getParameter("date");
  String test=request.getParameter("test");
  String statics=request.getParameter("statics");
  String customer=request.getParameter("customer");
  String typeoftest=request.getParameter("typeoftest");
  String userid=request.getParameter("userid");
  String t300 = "t300";
  if(userid==null){
    userid=(String)session.getValue("userid");
```

```
  }
  else {
    session.putValue("userid",userid);
  }
  if(userid==null) { %>
  <jsp:forward page="/index.jsp" />
  <% } %>
<jsp:useBean id="internalhistoricalBean" class="beans.InternalHistoricalBean"
scope="session" />
<jsp:setProperty name="internalhistoricalBean" property="date" value="<%=date%>"  />
<jsp:setProperty name="internalhistoricalBean" property="customer" value="<%=customer%>"
/>
<jsp:setProperty name="internalhistoricalBean" property="material" value="<%=material%>"
/>
<jsp:setProperty name="internalhistoricalBean" property="test" value="<%=test%>" />
<jsp:setProperty name="internalhistoricalBean" property="statics" value="<%=statics%>"  />
<jsp:setProperty name="internalhistoricalBean" property="connStr" value="<%=connStr%>"  />
<jsp:setProperty name="internalhistoricalBean" property="typeoftest"
value="<%=typeoftest%>" />
  <HTML>
  <HEAD><TITLE> Customer Historical Data </TITLE> </HEAD>
  <BODY background="/images/sky.jpg">
<%
   if (date != null && typeoftest != null && material != null && test != null && statics != null &&
customer != null) {

     session.putValue("connStr",connStr);
     session.putValue("userid",userid);
     %>
     <H3> CUSTOMER HISTORICAL DATA </H3>
     <H3>TEST RESULTS FOR: <I><%=customer%> </I></H3>
     <H3>TEST TYPE: <I><%=test%> </I></H3>
     <H3>DATE: <I><%=date%> </I></H3>
     <%
     String staticsyes = "YES";
     String staticsno = "NO";
     if(statics.compareTo(staticsno) == 0) {
     %>
     <%= internalhistoricalBean.getResult()%>
     <% }
     else { %>
     <%= internalhistoricalBean.getResult()%>
     <%= internalhistoricalBean.getStatistics()%>
     <% } %>
     <HR>
     <IMG SRC="/images/Normativ8D1.gif">
     <A HREF="/technician/UseInternalHistoricalBean.jsp"><IMG
SRC="/images/BlurMet7D4.gif"></A>
     <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%  } else {
```

```
%>
<HTML>
<HEAD> <TITLE> Customer Historical Data </TITLE> </HEAD>
<H1><b> Customer Historical Data </b> </H1>
<H3><B>Enter a search condition:</B></H3>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<P><b>Customer Name:</b> <INPUT TYPE="TEXT" NAME="customer" SIZE=25
VALUE="<%%>">
<P>(Enter the customer name e.g Speedy, Eurolam, etc: first letter must be capital)
<HR>
<P><b>Select Date :</b> <INPUT TYPE="TEXT" NAME="date" SIZE=11 VALUE="<%%>">
<P>(Format: DD,MMM,YYYY e.g 01-JAN-2000, 23,SEP-2001, etc. The query will retrieve data
from this date up to now)
<HR><b>Select material type:  </b>
<INPUT TYPE="radio" NAME="material" VALUE="LAMINATE">LAMINATE
<INPUT TYPE="radio" NAME="material" VALUE="MULTILAYER-BOARD">MULTILAYER-
BOARD
<HR><b>Select type of test:  </b>
<INPUT TYPE="radio" NAME="test" VALUE="TRANSITION">TRANSITION
<INPUT TYPE="radio" NAME="test" VALUE="RESIDUAL">RESIDUAL
<P><INPUT TYPE="radio" NAME="test" VALUE="RESISTANCE">RESISTANCE
Test type: <INPUT TYPE="TEXT" NAME="typeoftest" SIZE=4 VALUE="<%=t300%>">
<P>(If Resistance is selected, enter the type of test: t300 or t288 or t260)
<P><HR>
<P><b>Statistical Summary: </b>
<INPUT TYPE="radio" NAME="statics" VALUE="YES">YES
<INPUT TYPE="radio" NAME="statics" VALUE="NO">NO
<P><HR>
<INPUT TYPE="submit" VALUE="SUBMIT">
<INPUT TYPE="reset" VALUE="RESET">
</FORM>

<IMG SRC="/images/Normativ8D1.gif">
<A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<% } %>
</BODY>
</HTML>
key-path:technical service representative
file: index.jsp
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <meta name="GENERATOR" content="Mozilla/4.72 [en] (X11; I; SunOS 5.6 sun4u)
[Netscape]">
  <title>Technical Service Representative Main Menu</title>
</head>
<body background="/images/sky.jpg">
 
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
```

```
<tr VALIGN=TOP>
<td WIDTH="100%">
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td>
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td COLSPAN="3"><a NAME="general"></a><b><font face="Arial, Helvetica"><font
size=+5>Technical Service Representative's  Main Menu
</font></font></b></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica"><font size=+2>Read Customer Request
</font></font></b>
<br><font size=+1>The Technical Service Representative
reads customers' request to release a final request
on which the laboratory technician will perform testing
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/techsrvrep/UseUpdateRequest.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
<font size=+2>Update Test Results</font></font></b>
<br><font size=+1>
The Technical Service Representative updates test results
by giving a technical advice and releasing the record for
the manager to review
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/techsrvrep/UseUpdateTestResult.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
```

```
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
<font size=+2>Historical Data</font></font></b>
<br><font size=+1>
The Technical Service Representative can retrieve
historical data of any customer for different types of
tests
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/techsrvrep/UseInternalHistoricalBean.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
<font size=+2>Exit</font></font></b>
<br><font size=+1>
Go back to main menu
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/index.html">Exit</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

</table>
</td>
</tr>
</table>
</td>
```

```
<td WIDTH="15"></td>

<td WIDTH="1"></td>

<td WIDTH="15"></td>

<td WIDTH="100"> </td>
</tr>
</table>
</body>
</html>
```

Key-path: Technical Service Representative
file: UseUpdateRequest.jsp

```
<%@ page import="java.sql.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String observations=request.getParameter("observations");
    String tg=request.getParameter("tg");
    String t300=request.getParameter("t300");
    String t288=request.getParameter("t288");
    String t260=request.getParameter("t260");
    String enthalpy=request.getParameter("enthalpy");
    String connStr=request.getParameter("connStr");
    if(connStr==null || userid==null) {
    connStr=(String)session.getValue("connStr");
    userid=(String)session.getValue("userid");
  } else {
    session.putValue("connStr",connStr);
    session.putValue("userid",userid);
  }
  if (connStr==null || userid==null) { %>
<jsp:forward page="/index.jsp" />
<%
  } %>
<jsp:useBean id="updateRequestBean" class="beans.UpdateRequestBean" scope="session"
/>
<jsp:setProperty name="updateRequestBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="updateRequestBean" property="userid"  />
<jsp:setProperty name="updateRequestBean" property="requestid"  />
<jsp:useBean id="finalRequestBean" class="beans.UpdateRequestBean" scope="session" />
<jsp:setProperty name="finalRequestBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="finalRequestBean" property="requestid" />
<jsp:setProperty name="finalRequestBean" property="userid" />
<jsp:setProperty name="finalRequestBean" property="observations" />
<jsp:setProperty name="finalRequestBean" property="tg" />
<jsp:setProperty name="finalRequestBean" property="t300" />
<jsp:setProperty name="finalRequestBean" property="t288" />
<jsp:setProperty name="finalRequestBean" property="t260" />
<jsp:setProperty name="finalRequestBean" property="enthalpy" />
```

```
<HTML>
  <HEAD>
    <TITLE>
       Read Customer Request
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1> Read Customer Request </H1>
<%
  if (observations != null) {
     if(connStr==null && userid==null && requestid==null){
     session.putValue("connStr",connStr);
     session.putValue("userid",userid);
     session.putValue("requestid",requestid); }
     %>
     <HR>
     <%= finalRequestBean.getUpdate()  %>
     <%= finalRequestBean.getUpdateTables() %>
     <H1>Select Another Request from the Table: </H1>
<%
     }
  Connection conn = DriverManager.getConnection(connStr,"ben28","benito29");
  Statement stmt = conn.createStatement();
  try{
    String sql= "SELECT r.request_id, r.customer_id,TO_CHAR(r.date_requested)," +
                "r.sample_id, r.material_type, r.nature_ofthe_problem" +
                " FROM request r, final_request fr " +
                " WHERE r.request_id=fr.request_id " +
                " AND fr.propertesting_toperform = ' ' " +
                " AND r.techsrvrep_id="'+userid+"'" +
                " AND fr.observations = ' '" +
                " ORDER BY r.date_requested" ;
    ResultSet rset = stmt.executeQuery(sql);
    if(rset.next()){
%>
<%@ page import="java.sql.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String techadvice=request.getParameter("techadvice");
    String diagnosis=request.getParameter("diagnosis");
    String connStr=request.getParameter("connStr");
    if(connStr==null) {
    connStr=(String)session.getValue("connStr");
  } else {
    session.putValue("connStr",connStr);
  }
  if (connStr==null) { %>
<jsp:forward page="/index.jsp"/>
<%
```

```
  } %>
<jsp:useBean id="updateTestResultBean" class="beans.UpdateTestResultBean"
scope="session" />
<jsp:setProperty name="updateTestResultBean" property="connStr" value="<%= connStr %>"
/>
<jsp:setProperty name="updateTestResultBean" property="userid"  />
<jsp:setProperty name="updateTestResultBean" property="requestid"  />
<jsp:useBean id="historicalUpdateBean" class="beans.UpdateTestResultBean"
scope="session" />
<jsp:setProperty name="historicalUpdateBean" property="connStr" value="<%= connStr %>"
/>
<jsp:setProperty name="historicalUpdateBean" property="diagnosis" />
<jsp:setProperty name="historicalUpdateBean" property="requestid" />
<jps:setProperty name="historicalUpdateBean" property="userid" />
<jsp:setProperty name="historicalUpdateBean" property="techadvice" />

<HTML>
  <HEAD>
    <TITLE>
       Update Test Results
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1> Update Test Results </H1>
<%
 Connection conn = DriverManager.getConnection(connStr,"ben28","benito29");
 Statement stmt = conn.createStatement();
 if (diagnosis != null ) {
     if(connStr==null && userid==null && requestid==null){
     session.putValue("connStr",connStr);
     session.putValue("userid",userid);
     session.putValue("requestid",requestid);
     }
     if(diagnosis == "" ) {
     %>
     <H1>Entries cannot be empty !</H1>
     <% }
     else {
     %>
    <%= historicalUpdateBean.getUpdate()  %>
    <H1>Scroll Down and Make Another Selection from the Table: </H1>

<% requestid = null;}}
 try{
   String sql= "SELECT DISTINCT r.request_id, r.customer_id,TO_CHAR(e_requested,'fmDD
Month YYYY')," +
                "r.sample_id, r.material_type, r.nature_ofthe_problem" +
                " FROM request r, final_request fr, historical h, testing t " +
                " WHERE r.request_id=fr.request_id " +
                " AND fr.request_id=h.historical_id " +
                " AND h.historical_id=t.historical_id " +
```

```
                " AND r.testing_status = 'TESTING COMPLETED' " +
                " AND h.diagnosis = ' ' " +
                " AND h.technical_advice = ' ' ";
    ResultSet rset = stmt.executeQuery(sql);
    if(rset.next()){
%>
   <TABLE BORDER=1 BGCOLOR="C0C0C0">
   <TH WIDTH=200 BGCOLOR="white"> <I> Request Id </I> </TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Customer Name </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Date </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Sample Id </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Material Type</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Description of The Problem</I></TH>
   <TR> <TD ALIGN=CENTER> <%= rset.getString(1)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(2)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(3)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(4)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(5)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(6)%></TD>
   </TR>
<%   while (rset.next()) {
%>
    <TR> <TD ALIGN=CENTER> <%= rset.getString(1) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(2) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(3) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(4) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(5) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(6) %></TD>
     </TR>
<% }
%>
    </TABLE>
<%  }
   else {
%>
  <P> Sorry, the query returned no rows! </P>
  <P> There is no customer's requests for reviewing right now !</P>
<%
   }
   rset.close();
   stmt.close();
  } catch (SQLException e) {
    out.println("<P>" + "There was an error doing the query:");
    out.println ("<PRE>" + e + "</PRE> \n <P>");
    }
    finally { %>
<H1><P><B>Enter a search condition:</B></P></H1>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<HR>
<P><b>Request id:</b> <INPUT TYPE=text NAME="requestid" SIZE=6 VALUE="<%%>" >
```

```
<P>(Enter the request id e.g: 2, 23, 24, 123, etc.)
<HR>
<P><INPUT TYPE=submit VALUE="SUBMIT">
<INPUT TYPE=reset VALUE="RESET">
</FORM>
 <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<%  } %>

<%
   if (requestid != null) {
 conn = DriverManager.getConnection(connStr,"ben28","benito29");
 stmt = conn.createStatement();
 String tg = null;
 String enthalpyCheck = null;
 String t300min = null;
 String t288min = null;
 String t260min = null;
   try {

   String query = "SELECT tr.tg FROM request r, final_request fr, historical h, " +
                " testing t, transition tr " +
                " WHERE r.request_id=fr.request_id " +
                " AND fr.request_id=h.historical_id " +
                " AND t.historical_id=h.historical_id " +
                " AND t.testing_id=tr.testing_id " +
                " AND t.test_type='TRANSITION'" +
                " AND t.historical_id='" +requestid+ "'";
   ResultSet  rset = stmt.executeQuery(query);
   if(rset.next())
   tg = rset.getString(1);
        query = "SELECT rc.enthalpy FROM request r, final_request fr, historical h, " +
              " testing t, residual_cure rc " +
               " WHERE r.request_id=fr.request_id " +
               " AND fr.request_id=h.historical_id " +
               " AND t.historical_id=h.historical_id " +
               " AND t.testing_id=rc.testing_id " +
               " AND t.test_type='RESIDUAL'" +
               " AND t.historical_id='" +requestid+ "'";
           rset = stmt.executeQuery(query);
   if(rset.next())
   enthalpyCheck = rset.getString(1);

  query = "SELECT res.t300_min, res.t288_min, res.t260_min " +
        " FROM request r, final_request fr, historical h, "+
               " testing t, resistance res " +
               " WHERE r.request_id=fr.request_id " +
               " AND fr.request_id=h.historical_id " +
               " AND t.historical_id=h.historical_id " +
               " AND t.testing_id=res.testing_id " +
               " AND t.test_type='RESISTANCE'" +
```

```
            " AND t.historical_id='" +requestid+ "'";

          rset = stmt.executeQuery(query);
   if(rset.next())
   {
   t300min = rset.getString(1);
   t288min = rset.getString(2);
   t260min = rset.getString(3);
   }
   stmt.close();
   rset.close();
      }

    catch (SQLException e) {
        out.println("<P>" + "This was an error doing the query:");
       out.println ("<PRE>" + e + "</PRE> \n <P>");
     }

%>
    <H1>Sample Information</H1>
    <%= updateTestResultBean.getResultInfo() %>
 <% if(tg != null) { %>
    <H1>Transition Test Results</H1>
    <%= updateTestResultBean.getResultTransition() %>
 <% }  if (t300min != null) { %>

    <H1>Resistance Test Results</H1>
    <%= updateTestResultBean.getResultT300() %>
 <% } if (t288min != null) { %>
    <H1> T288 Test Results </H1>
    <%= updateTestResultBean.getResultT288() %>
 <% } if (t260min != null) { %>
    <H1> T260 Test Results </H1>
    <%= updateTestResultBean.getResultT260() %>
 <% } if (enthalpyCheck != null) { %>
    <H1>Residual Cure Test Results</H1>
    <%= updateTestResultBean.getResultResidual() %>
 <% } %>
    <FORM METHOD=get>
    <font face="Arial, Helvetica"><font size=+1>
    <P><b>Request Id:</b><INPUT TYPE=text NAME="requestid" SIZE=6
VALUE="<%=requestid%>" >
    <P>(Enter the request id)
    <HR><b>Diagnosis:</b><P><TEXTAREA NAME="diagnosis" SIZE=400
VALUE="<%=diagnosis%>"  ROWS=3 COLS=65 WRAP></TEXTAREA>
    <P>(Enter the diagnosis based on the test results and information pertaining to the sample)
    <HR><b>Technical Advice:<b><P><TEXTAREA NAME="techadvice" SIZE=400
VALUE="<%=techadvice%>" ROWS=3 COLS=65 WRAP></TEXTAREA>
    <P>(Enter your technical advice to solve the problem)
    <HR>
   <P><INPUT TYPE=submit VALUE="SUBMIT">
```

```
    <INPUT TYPE=reset VALUE="RESET">
    </FORM>
 <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>
<%
 }
     %>
</BODY>
</HTML>
```

file: UseInternalHistoricalBean.jsp

```
<%
  String material=request.getParameter("material");
  String date=request.getParameter("date");
  String test=request.getParameter("test");
  String statics=request.getParameter("statics");
  String connStr=request.getParameter("connStr");
  String customer=request.getParameter("customer");
  if (connStr==null){
    connStr=(String)session.getValue("connStr");
   }else {
     session.putValue("connStr",connStr);
   }
  if (connStr==null) { %>
  <jsp:forward page="/index.jsp" />
  <% } %>
<jsp:useBean id="internalhistoricalBean" class="beans.InternalHistoricalBean"
scope="session" />
<jsp:setProperty name="internalhistoricalBean" property="date" value="<%=date %>"  />
<jsp:setProperty name="internalhistoricalBean" property="customer" value="<%=customer
%>" />
<jsp:setProperty name="internalhistoricalBean" property="material" value="<%=material%>"
/>
<jsp:setProperty name="internalhistoricalBean" property="test" value="<%=test %>" />
<jsp:setProperty name="internalhistoricalBean" property="statics" value="<%=statics %>"  />
<jsp:setProperty name="internalhistoricalBean" property="connStr" value="<%=connStr%>"  />
  <HTML>
  <HEAD><TITLE> Customer Historical Data </TITLE> </HEAD>
  <BODY background="/images/sky.jpg">
<%
   if (date != null) {

     session.putValue("connStr",connStr);
     %>
    <H3> CUSTOMER HISTORICAL DATA </H3>
    <H3>TEST RESULTS FOR: <I><%=customer%> </I></H3>
    <H3>TEST TYPE: <I><%=test%> </I></H3>
    <H3>DATE: <I><%=date%> </I></H3>
    <%
    String staticsyes = "YES";
    String staticsno = "NO";
```

```
     if(statics.compareTo(staticsno) == 0) {
    %>
    <%= internalhistoricalBean.getResult()%>
     <% }
     else { %>
    <%= internalhistoricalBean.getResult()%>
    <%= internalhistoricalBean.getStatistics()%>
    <% } %>
    <HR>
    <IMG SRC="/images/Normativ8D1.gif">
    <A HREF="/technician/UseInternalHistoricalBean.jsp"><IMG
SRC="/images/BlurMet7D4.gif"></A>
    <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%  } else {
%>
<HTML>
<HEAD> <TITLE> Customer Historical Data </TITLE> </HEAD>
<H1><b> Customer Historical Data </b> </H1>
<H3><B>Enter a search condition:</B></H3>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<P><b>Customer Name:</b> <INPUT TYPE="TEXT" NAME="customer" SIZE=25
VALUE="<%%>">
<P>(Enter the customer name e.g Speedy, Eurolam, etc: first letter must be capital)
key-path: manager
file: index.jsp
<%@page import="java.sql.*" %>
<%
  String userid=request.getParameter("useid");
  String connStr=request.getParameter("connStr");
  if (connStr==null || userid==null) {
    connStr=(String)session.getValue("connStr");
    userid=(String)session.getValue("userid");
  }
  session.putValue("connStr", connStr);
  session.putValue("userid", userid);
  if(connStr==null || userid==null) { %>
<jsp:forward page="/index.jps" />
<% } %>
<H1>Welcome <I> <%=userid%> </I> !</H1>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
  <meta name="GENERATOR" content="Mozilla/4.72 [en] (X11; I; SunOS 5.6 sun4u)
[Netscape]">
  <title>Manager Main Thing</title>
</head>
<body background="/images/sky.jpg">
 
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
```

```
<tr VALIGN=TOP>
<td WIDTH="100%">
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td>
<table BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH="100%" >
<tr>
<td COLSPAN="3"><a NAME="general"></a><b><font face="Arial, Helvetica"><font
size=+5>Manager Main Menu
</font></font></b></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica"><font size=+2>Read Customer Request
</font></font></b>
<br><font size=+1>The manager reads final customer request
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/manager/UseReadRequestBeansqlj.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>

<tr>
<td WIDTH="35"></td>
<td COLSPAN="2"><b><font face="Arial, Helvetica">
<font size=+2>Update Test Results</font></font></b>
<br><font size=+1>
The manager updates test results by giving a final technical advice
</font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"><spacer TYPE="HORIZONTAL" SIZE="35"><font size=-1>
<a href="/manager/UseFinalResultBean.jsp">Take a look</a></font></td>
</tr>
<tr>
<td WIDTH="35"> </td>
<td WIDTH="35"></td>
<td WIDTH="100%"></td>
</tr>
```

Key-path: Manager
file: UseReadRequestBean.jsp

```
<%@ page import="java.sql.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String connStr=request.getParameter("connStr");
    if(connStr==null) {
    connStr=(String)session.getValue("connStr");
  } else {
    session.putValue("connStr",connStr);
  }
  if (connStr==null) { %>
<jsp:forward page="/index.jsp" />
<%
  } %>
<jsp:useBean id="readRequestBean" class="beans.ReadRequestBean" scope="session" />
<jsp:setProperty name="readRequestBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="readRequestBean" property="userid"  />
<jsp:setProperty name="readRequestBean" property="requestid"  />
<HTML>
  <HEAD>
    <TITLE>
       Read Customer Request
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1> Read Customer Request </H1>
<%
 Connection conn = DriverManager.getConnection(connStr,"ben28","benito29");
 Statement stmt = conn.createStatement();
 try{
    String sql= "SELECT r.request_id, r.customer_id,TO_CHAR(r.date_requested)," +
                "r.sample_id, r.material_type, r.nature_ofthe_problem, r.testing_status" +
                " FROM request r " +
                " WHERE r.testing_status <> 'COMPLETED'";



    ResultSet rset = stmt.executeQuery(sql);
    if(rset.next()){
%>
   <TABLE BORDER=1 BGCOLOR="C0C0C0">
   <TH WIDTH=200 BGCOLOR="white"> <I> Request Id </I> </TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Customer Name </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Date Requested</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Sample Id </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Material Type</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Description of The Problem</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Testing Status</I></TH>
```

```
   <TR> <TD ALIGN=CENTER> <%= rset.getString(1)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(2)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(3)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(4)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(5)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(6)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(7)%></TD>
   </TR>
<%   while (rset.next()) {
%>
    <TR> <TD ALIGN=CENTER> <%= rset.getString(1) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(2) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(3) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(4) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(5) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(6) %></TD>
        <TD ALING=CENTER> <%= rset.getString(7) %></TD>
     </TR>
<% }
%>
    </TABLE>
<%  }
   else {
%>
  <P> Sorry, the query returned no rows! </P>
<%
   }
   rset.close();
   stmt.close();
  } catch (SQLException e) {
    out.println("<P>" + "There was an error doing the query:");
    out.println ("<PRE>" + e + "</PRE> \n <P>");
    }
%>

<%
   if (requestid != null) { %>
     <H3>Search Results</H3>
     <%=  readRequestBean.getResult() %>

<% } %>

<P><B>Enter a search condition:</B></P>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<HR>
<P><b>Request id:</b> <INPUT TYPE=text NAME="requestid" SIZE=6 VALUE="<%%>" >
<P>(Enter request id e.g 23, 12, 123, etc. Must be a valid id)
<HR>
<INPUT TYPE=submit VALUE="SUBMIT">
<INPUT TYPE=reset VALUE="RESET">
```

```
</FORM>
  <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

</BODY>
</HTML>
Key-path: Manager
file: UseUpdateFinalResultBean.jsp
<%@ page import="java.sql.*" %>
<%@ page import="java.util.*" %>
<%@ page import="javax.mail.*" %>
<%@ page import="javax.mail.internet.*" %>
<%@ page import="javax.activation.*" %>
<%@ page import="java.util.Properties.*" %>
<%
    String userid=request.getParameter("userid");
    String requestid=request.getParameter("requestid");
    String mgradv=request.getParameter("mgradv");
    String connStr=request.getParameter("connStr");
    String email=request.getParameter("email");
    String to=request.getParameter("to");
    String from=request.getParameter("from");
    String subject=request.getParameter("subject");
    String message=request.getParameter("message");
    if(connStr==null) {
    connStr=(String)session.getValue("connStr");
  } else {
    session.putValue("connStr",connStr);
  }
  if (connStr==null) { %>
<jsp:forward page="/index.jsp"/>
<%
  } %>
<jsp:useBean id="updatefinalresultBean" class="beans.UpdateFinalResultBean"
scope="session" />
<jsp:setProperty name="updatefinalresultBean" property="connStr" value="<%= connStr %>"
/>
<jsp:setProperty name="updatefinalresultBean" property="userid"  />
<jsp:setProperty name="updatefinalresultBean" property="requestid"  />
<jsp:setProperty name="updatefinalresultBean" property="customerid" />
<jsp:setProperty name="updatefinalresultBean" property="mgradv" />
<jsp:useBean id="historicalupdateBean" class="beans.UpdateFinalResultBean"
scope="session" />
<jsp:setProperty name="historicalupdateBean" property="connStr" value="<%= connStr %>" />
<jsp:setProperty name="historicalupdateBean" property="mgradv" />
<jsp:setProperty name="historicalupdateBean" property="requestid" />
<jsp:setProperty name="historicalupdateBean" property="userid" />
<HTML>
  <HEAD>
    <TITLE>
      Update Test Results
```

```
    </TITLE>
  </HEAD>
<body background="/images/sky.jpg">
<H1> Update Test Results </H1>
<%
 Connection conn = DriverManager.getConnection(connStr,"ben28","benito29");
 Statement stmt = conn.createStatement();
   if(mgradv != null)
    {
    %>
   <H1>Your Final Technical Advice for request id: <%=requestid%> has been submitted</H1>
   <p><H1> and the Test Results have been released for the customer !</H1>
   <p><H1> scroll down and notify the customer by sending an e-mail </H1>
   <%
     }
   if(message != null)
   {
   %>
   <H1>An e-mail to the customer has been sent !</H1>
   <H3>Make another selection from the table: </H3>

   <% }
 try{
   String sql= "SELECT DISTINCT r.request_id, r.customer_id,TO_CHAR(r.mfg_date)," +
               "r.sample_id, r.material_type, r.nature_ofthe_problem" +
               " FROM request r, final_request fr, historical h, testing t " +
               " WHERE r.request_id=fr.request_id " +
               " AND fr.request_id=h.historical_id " +
               " AND h.historical_id=t.historical_id " +
               " AND r.testing_status= 'TEST RESULTS REVIEWED BY TSR'";

   ResultSet rset = stmt.executeQuery(sql);
   if(rset.next()){
%>
   <TABLE BORDER=1 BGCOLOR="C0C0C0">
   <TH WIDTH=200 BGCOLOR="white"> <I> Request Id </I> </TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Customer Name </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Date Requested</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Sample Id </I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Material Type</I></TH>
   <TH WIDTH=200 BGCOLOR="white"> <I> Description of The Problem</I></TH>
   <TR> <TD ALIGN=CENTER> <%= rset.getString(1)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(2)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(3)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(4)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(5)%></TD>
    <TD ALIGN=CENTER> <%= rset.getString(6)%></TD>
   </TR>
<%   while (rset.next()) {
%>
    <TR> <TD ALIGN=CENTER> <%= rset.getString(1) %></TD>
```

```
        <TD ALIGN=CENTER> <%= rset.getString(2) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(3) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(4) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(5) %></TD>
        <TD ALIGN=CENTER> <%= rset.getString(6) %></TD>
      </TR>
<% }
%>
    </TABLE>
<%  }
   else {
%>
  <P> Sorry, There are no requests to be updated! Try again later </P>
<%
   }
   rset.close();
   stmt.close();
  } catch (SQLException e) {
    out.println("<P>" + "There was an error doing the query:");
    out.println ("<PRE>" + e + "</PRE> \n <P>");
    }
    finally { %>
<P><B>Enter a search condition:</B></P>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<HR>
<P><b>Request id:</b> <INPUT TYPE=text NAME="requestid" SIZE=6 VALUE="<%%>" >
<P>(Enter a valid request id e.g 12, 234, 12, etc)
<HR>
<P><INPUT TYPE=submit VALUE="SUBMIT">
<INPUT TYPE=reset VALUE="RESET">
</FORM>
 <IMG SRC="/images/Normativ8D1.gif">
   <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>



<%   } %>

<%
   if (requestid != null) {
  conn = DriverManager.getConnection(connStr,"ben28","benito29");
  stmt = conn.createStatement();
  String tg = null;
  String enthalpyCheck = null;
  String t300min = null;
  String t288min = null;
  String t260min = null;
    try {

    String query = "SELECT tr.tg FROM request r, final_request fr, historical h, " +
```

```
            " testing t, transition tr " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=tr.testing_id " +
            " AND t.test_type='TRANSITION'" +
            " AND t.historical_id='" +requestid+ "'";
  ResultSet  rset = stmt.executeQuery(query);
  if(rset.next())
  tg = rset.getString(1);
      query = "SELECT rc.enthalpy FROM request r, final_request fr, historical h, " +
            " testing t, residual_cure rc " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=rc.testing_id " +
            " AND t.test_type='RESIDUAL'" +
            " AND t.historical_id='" +requestid+ "'";
          rset = stmt.executeQuery(query);
  if(rset.next())
  enthalpyCheck = rset.getString(1);
query = "SELECT res.t300_min, res.t288_min, res.t260_min " +
       " FROM request r, final_request fr, historical h, "+
            " testing t, resistance res " +
            " WHERE r.request_id=fr.request_id " +
            " AND fr.request_id=h.historical_id " +
            " AND t.historical_id=h.historical_id " +
            " AND t.testing_id=res.testing_id " +
            " AND t.test_type='RESISTANCE'" +
            " AND t.historical_id='" +requestid+ "'";

          rset = stmt.executeQuery(query);
  if(rset.next())
  {
  t300min = rset.getString(1);
  t288min = rset.getString(2);
  t260min = rset.getString(3);
  }
  stmt.close();
  rset.close();
      }

    catch (SQLException e) {
        out.println("<P>" + "This was an error doing the query:");
       out.println ("<PRE>" + e + "</PRE> \n <P>");
     } %>

   <H1>Sample Information</H1>
   <%= updatefinalresultBean.getResultInfo() %>
<% if(tg != null) { %>
   <H1>Transition Test Results</H1>
```

```
    <%= updatefinalresultBean.getResultTransition() %>
 <% }  if (t300min != null) { %>

    <H1>T300 Test Results</H1>
    <%= updatefinalresultBean.getResultT300() %>
 <% } if (t288min != null) { %>
    <H1> T288 Test Results </H1>
    <%= updatefinalresultBean.getResultT288() %>
 <% } if (t260min != null) { %>
    <H1> T260 Test Results </H1>
    <%= updatefinalresultBean.getResultT260() %>
 <% } if (enthalpyCheck != null) { %>
    <H1>Residual Cure Test Results</H1>
    <%= updatefinalresultBean.getResultResidual() %>
 <% }  %>
    <H1> Technical Service Representative Advice </H1>
    <%= updatefinalresultBean.getResultAdvice() %>

    <FORM METHOD=get>
    <font face="Arial, Helvetica"><font size=+1>
    <P><b>Request Id:</b> <INPUT TYPE=text NAME="requestid" SIZE=6
VALUE="<%=requestid%>" >
    <P>(Enter a valid request id. A number selected from the list above)
    <HR>
    <HR><b>Final Technical Advice:</b><P><TEXTAREA NAME="mgradv" SIZE=400
ROWS=3 COLS=65 WRAP>Diagnosis?</TEXTAREA>
   <P>(Enter a final technical advice considering test results and technical service
representative advice)
   <HR>
   <P><INPUT TYPE=submit VALUE="SUBMIT">
    <INPUT TYPE=reset VALUE="RESET">
    </FORM>
 <IMG SRC="/images/Normativ8D1.gif">
  <A HREF="index.jsp"><IMG SRC="/images/Normativ5D1.gif"></A>

<%
 }
  if (mgradv != null) {
     if(connStr==null && userid==null && requestid==null){
     session.putValue("connStr",connStr);
     session.putValue("userid",userid);
     session.putValue("requestid",requestid);
     }
     conn = DriverManager.getConnection(connStr,"ben28","benito29");
     stmt = conn.createStatement();
     try {
        String query ="SELECT customer_id "+
                 "FROM request " +
                 "WHERE request_id='"+requestid+"'";
        ResultSet  rset = stmt.executeQuery(query);
        String user_id= null;
```

```
        if(rset.next()){
        user_id = rset.getString(1);
        }
             query ="SELECT email " +
                  "FROM customer " +
                  "WHERE user_id='"+user_id+"'";
             rset = stmt.executeQuery(query);
         if(rset.next()) {
           email = rset.getString(1);
         }

        stmt.close();
        rset.close();
        }

       catch (SQLException e) {
          out.println("<P>" + "This was an error doing the query:");
          out.println ("<PRE>" + e + "</PRE> \n <P>");
       }
       finally {  %>
     <%=historicalupdateBean.getUpdate() %>
     <%
       }
} %>
<%
  if (email != null) {
String Thesender= "oracle@bengp.ias.csusb.edu";
String Defaultsubject= "Testing for  request id: " +requestid+ " has been completed";
String Defaultmessage= "You submitted a testing request and it " +
      " has been completed, access the system " +
      " to see test results and technical advice ";
%>
<H1>Sending Email to Notify Customer</H1>
<FORM METHOD=get>
<font face="Arial, Helvetica"><font size=+1>
<p><b>From: </b> <INPUT TYPE="text" NAME="from" SIZE=50 VALUE="<%=Thesender%>"
>
<HR>
<p><b>To: </b> <INPUT TYPE="text" NAME="to" SIZE=50 VALUE="<%=email%>" >
<HR>
<p><b>Subject: </b><INPUT TYPE="text" NAME="subject" SIZE=50
VALUE="<%=Defaultsubject%>" >
<HR><p><b>Message: </b>[if appropiate, edit the message]<P><TEXTAREA
NAME="message" SIZE=400 ROWS=3 COLS=65 WRAP>You submitted a testing request
and it has been completed, access the system to see test results and technical
advice</TEXTAREA>
<P>
<INPUT TYPE="submit" VALUE="Submit Email"  >
</FORM>
<% }
    if(message != null)
```

```
    {
 try {
%>
    <%= runPostMail(to, from, subject, message) %>
<%    } catch(MessagingException e) {
        out.println("<P>" + "There was an error ");
     } }
%>
<%!
    private String runPostMail(String to, String from, String subject, String message)
                throws MessagingException{
    boolean debug = true;
      Properties props = new Properties();
      props = System.getProperties();
      props.put("mail.smtp.host", "gallium.ias.csusb.edu");
    Session session = Session.getInstance(props);
    session.setDebug(debug);

    Message msg = new MimeMessage(session);

    InternetAddress addressFrom = new InternetAddress(from);
    msg.setFrom(addressFrom);

    String[] recipients = new String[1];
     recipients[0] =to;

     InternetAddress[] addressTo = new InternetAddress[recipients.length];
     for(int i = 0; i < recipients.length; i++)
     {
       addressTo[i] = new InternetAddress(recipients[i]);
     }

    msg.setRecipients(Message.RecipientType.TO, addressTo);
    msg.addHeader("MyHeaderName", "myHeaderValue");
    msg.setSubject(subject);
    msg.setContent(message, "text/plain");
    Transport.send(msg);
    return (" ");
}
 %>
</BODY>
</HTML>
```

REFERENCES

[1] Michael Blaha and William Premerlani, Object-Oriented Modeling and Design for Database Applications, First Edition, Prentice Hall,1998.

[2] Ramez Elmasri and Shamkant B. Navathe, Fundamentals Of Database Systems, Second Edition, Addison-Wesley Publishing Company,1994.

[3] Taylor Art , JDBC Developer's Resource, Second Edition, Prentice Hall 1999.

[4] Horton Ivor, Beginning Java 2, JDK 1.3 Edition, Wrox Press, 2000.

[5] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

[6] Fowler & Scott, UML Distilled, A Brief Guide to Standard Object Modeling Language,1999.

[7] Ched Guiles, Everything You ever wanted to know about laminates, 8th Edition,@Arlon Inc. 2000.

[8] IPC(Institute for Printed Circuits:Association Connecting Electronics Industries)-TM-650, Test Methods Manual.

[9] IPC-TM-4101, Specifications for Base Materials for Rigid Boards and Multilayer Printed Boards.

[10] IPC-TM-1710, OEM Standard for Printed Board Manufacturers Qualification Profile.

[11] Merriam-Webster's Collegiate Dictionary. web-address:http://www.com/cgibin/dictionary.

[12] Carnell John, Bjarki Hólm & Horton Ann, Oracle8i Application Programming with Java, PL/SQL and XML,First Edition. Wrox Press, 2000.

[13] Welsh Matt, Kalle Matthias, Dalmeiher Dalle & Kaufman Lar, Third Edition, O'Reilly & Associates,1999.

[14] Zobel Justin, Writing for Computer Science. Springer, 1998.

[15] Castro Elizabeth, HTML for the World Wide Web, Fourth Edition. Peachpit Press, 2000.

[16] Spencer Paul, Professional XML Design and Implementation, First Edition. Wrox Press, 2000.

[17] Oracle8i Installation Guide, Release 3(8.1.7) for Linux Intel. @ Oracle, December 2000.

[18] Introduction to Oracle: SQL and PL/SQL, Production Volume 1 and 2, Student Guide. @ Oracle, April 1998.

[19] Oracle: Database Administration, Production 1.0 Volume 1 and 2, Student Guide. @ Oracle, March March 1998.