CHARM WebIF

Nirmal G. Tiwari
Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/gradreports

Part of the Computer Sciences Commons

Recommended Citation
Tiwari, Nirmal G., "CHARM WebIF" (2011). All Graduate Plan B and other Reports. 47.
https://digitalcommons.usu.edu/gradreports/47
CHARM Web-Interface

by

Nirmal G. Tiwari

A report submitted in partial fulfillment of the requirements for the degree

of

MASTER OF SCIENCE

in

Computer Science

Approved:

Stephen Clyde
Major Professor

Scott Cannon
Committee Member

Curtis Dyreson
Committee Member

UTAH STATE UNIVERSITY
Logan, Utah

2011
ABSTRACT

CHARM WebIF

by

Nirmal G Tiwari
Utah State University, 2011

Major Professor: Dr. Stephen Clyde
Department: Computer Science

This project presents a web-based interface for Child Health Advanced Record Management System (CHARM) users that provides easy and fast access to child information from multiple data sources. This project’s overarching goal is to integrate child information from different participating programs under a single interface to allow any CHARM user (including users not belonging to any particular participating program) to view information from different participating programs. To that end, the interface allows any registered CHARM user to access child information from multiple data sources without the user being a registered member of any particular participating program. Thus, the user can access information from a universal portal without the need of registering with any specific participating program. The user interfaces were designed and customized to make interaction with the CHARM WebIF convenient and user-friendly. This report discusses the application of the CHARM WebIF in the CHARM environment along with its user-friendly interface design.

(70 pages)
ACKNOWLEDGMENTS

I would like to thank my major professor, Dr. Stephen Clyde. His support and advice made successful completion of this project possible. His knowledge and expertise in GUI design inspired me to take on this project. I would also like to thank my committee members, Dr. Scott Cannon and Dr. Curtis Dyreson, for their support and valuable suggestions. I would like to thank my colleague Lan for her support and help throughout this project.

I would especially like to thank Keyur for giving me all the love and support for successfully completing all my academic pursuits.

Finally, I would like to thank my parents and my brother, without whom I could have never achieved anything. A special thanks to my would-be mother-in-law and father-in-law for their love and appreciation.

Nirmal G. Tiwari
# TABLE OF CONTENTS

ABSTRACT ..................................................................................................................

ACKNOWLEDGMENTS .................................................................................................

LIST OF FIGURES .........................................................................................................

CHAPTER

CHAPTER 1 .................................................................................................................... 1

INTRODUCTION ............................................................................................................. 1

CHAPTER 2 ..................................................................................................................... 4

BACKGROUND .............................................................................................................. 4

2.1 OVERVIEW .............................................................................................................. 4
2.2 CHARM ...................................................................................................................... 4
2.3 NEED FOR A CHARM WEBIF IN THE CHARM ENVIRONMENT ......................... 6

CHAPTER 3 ..................................................................................................................... 7

SYSTEM ANALYSIS ...................................................................................................... 7

3.1 OVERVIEW .............................................................................................................. 7
3.2 FUNCTIONAL REQUIREMENTS .............................................................................. 9
3.3 NON-FUNCTIONAL REQUIREMENTS ................................................................. 13
3.4 CONTEXT ANALYSIS ............................................................................................ 14
3.5 USER GOALS ......................................................................................................... 16
3.6 STRUCTURAL ANALYSIS ...................................................................................... 19

CHAPTER 4 ..................................................................................................................... 22

ARCHITECTURAL DESIGN ............................................................................................. 22

4.1 OVERVIEW .............................................................................................................. 22
4.2 USER INTERFACE DESIGN ................................................................................. 23
4.3 APPLICATION LAYER DESIGN ........................................................................... 25
4.4 DATABASE DESIGN ................................................................................................ 27
4.5 SYSTEM INTERACTION .......................................................................................... 29
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
</tr>
<tr>
<td>3.1</td>
</tr>
<tr>
<td>3.2</td>
</tr>
<tr>
<td>3.3</td>
</tr>
<tr>
<td>3.4</td>
</tr>
<tr>
<td>3.5</td>
</tr>
<tr>
<td>3.6</td>
</tr>
<tr>
<td>3.7</td>
</tr>
<tr>
<td>4.1</td>
</tr>
<tr>
<td>4.2</td>
</tr>
<tr>
<td>4.3</td>
</tr>
<tr>
<td>4.4</td>
</tr>
<tr>
<td>4.5</td>
</tr>
<tr>
<td>4.6</td>
</tr>
<tr>
<td>5.1</td>
</tr>
<tr>
<td>5.2</td>
</tr>
<tr>
<td>5.3</td>
</tr>
<tr>
<td>5.4</td>
</tr>
<tr>
<td>5.5</td>
</tr>
<tr>
<td>5.6</td>
</tr>
</tbody>
</table>
5.7 A screenshot of a child search .................................................................44
5.8 A screenshot of the select query page ......................................................46
5.9 A screenshot of the query results page ....................................................47
6.1 A screenshot of the password verification page ......................................51
6.2 A screenshot of the edit information page .............................................52
6.3 A screenshot of the search result page....................................................54
6.4 A screenshot of the view user information page ....................................55
6.5 A screenshot of the approve user page ....................................................56
CHAPTER 1
INTRODUCTION

The Child Health Advance Record Management System (CHARM) is part of the Utah Department of Health’s (UDOH) data integration effort. In general, it is an information broker that allows independent public health care programs to exchange data regarding children who are currently receiving or have received public health services. CHARM helps both public and private health care professionals by providing real-time access to the children’s health information. This information includes demographic information, such as, individual and family health risks, newborn heel-stick screening results, hearing-screening results, and immunization histories.

Until recently, all CHARM users, e.g., physicians, clinicians, and public health program personnel, have accessed child-health record through an existing information system, called *participating program*. Participating programs include Utah State Immunization Information System (USIIS), Vital Statistics (VS), Newborn Heel-stick Screening (NBS LIMS), Early Hearing Detection and Intervention (HiTrack), Part C Early Intervention (BTOTS), and the office of Recovery Services (ORS). Unfortunately, there are many potential CHARM users that do not have access to any of these systems. Specifically, a substantial number of pediatricians do not have access to any existing participating program and would benefit significantly in being able to see the results of heel-stick and hearing screenings as soon as they are available. Currently, to obtain newborn screening results, many pediatricians must call state offices and request patient records to be sent via facsimile. This process can be problematic and is prone to delays, particularly on weekends or holiday when state offices are closed.
This project describes a stand-alone web-based interface for CHARM that addresses the needs of potential CHARM users who do not have access through a participating program. The CHARM web-based interface, or WebIF for short, allows a CHARM user to access information from multiple participating programs, in accordance with security and confidentiality policies, without actually being directly associated with any particular participating program. Additionally, the CHARM WebIF provides facilities for user registration, user management, and account management. The CHARM WebIF provides single sign-on functionality, which allows the user to access data from multiple participating programs without the need of re-authentication by those participating programs. Moreover, it provides confidential searching of child and role-based access to queries associated with participating programs, thereby integrating the functionalities offered by different participating programs without the need of authenticating to those participating programs.

To provide some background and set the stage for the system’s requirements, Chapter 2 gives a brief overview of CHARM with a focus on the context for the WebIF. Chapter 3 discusses the user goals for the CHARM WebIF, provides a structural analysis, and describes the system’s functional and non-functional requirements. Chapter 4 then explains the design of the CHARM WebIF both in terms of the application-layer components and the database structures. This chapter also provides a description of the flow of the entire process. Chapter 5 discusses the interface design and organization and describes in detail the design principles used to build the web-based user interfaces. Chapter 6 provides a list of complete features of the CHARM WebIF along with the user
manuals for both a general user and an administrator. Chapter 7 describes the technology used in the implementation of CHARM WebIF. Chapter 8 provides information about the test cases conducted, as well as the general testing approaches used. Finally, Chapter 9 provides a summary of the project and suggests future enhancements.
CHAPTER 2
BACKGROUND

2.1 OVERVIEW

This chapter provides a brief overview of CHARM and some of its participating programs (PPs), thus establishing both the context of and need for the CHARM WebIF.

2.2 CHARM

CHARM is an information broker that allows independent health-care programs to exchange data about children who are currently receiving or have received public health services. Different health care programs that are associated with CHARM include Vital Statistics (VS), Utah Statewide Immunization Information System (USIIS), Early Hearing Detection and Invention (EHDI), New Born Screening (NBS) and BTOTS (Baby and Toddler Tracking System).

As shown in Figure 2.1, CHARM’s architecture consists of a CHARM server, a CHARM agent, and a participating program. When a particular PP requires data from CHARM, it submits a request for that data to the CHARM server via its own dedicated agent. That particular agent is responsible for mapping the PP’s specific data types and identifiers to CHARM’s specific data types and identifiers. The query is then transferred to the CHARM server, which in turn handles the request based on the most appropriate strategy for the requested type of data.
The request may require retrieval of data from multiple PPs, which is done by asking their agents for necessary data. Those agents then retrieve data from the PP and return it to the CHARM server. The CHARM server computes a final query result and returns it to the calling agent, which then converts the CHARM-specific data types and identities to PP-specific data types and identities.

The CHARM server maintains a CORE database that is used to match persons based on their demographic information. Other than the CORE database, each agent maintains a small identity-mapping database for correlating PP-specific identities with the internal CHARM identities.
2.3 Need for a CHARM WebIF in the CHARM Environment

The data from different participating programs is accessible to users who have registered with the respective participating programs. However, there are many CHARM users that do not have access to any of the existing systems. Accessing child health data is crucial for CHARM users. Specifically, a substantial number of CHARM users are medical professionals who are not registered with any specific PP. For such users, knowing child health-related information for their patients can be extremely important and beneficial as it plays a significant role in treatment and medication. The availability of health information is vital and must be accessible readily. To address such needs of a general CHARM user, we introduce the CHARM WebIF. The CHARM WebIF addresses the needs of users that do not belong to any of the user groups of any existing programs and to whom the availability and accessibility of data from multiple PPs is extremely important. The CHARM WebIF provides access to information from multiple PPs without the user actually being directly associated with any particular PP. The CHARM WebIF also integrates the functionality offered by different participating program, thereby providing access to information from multiple PPs just by authenticating once to the CHARM WebIF and avoiding the need of separate authentication to different PPs.
CHAPTER 3
SYSTEM ANALYSIS

3.1 OVERVIEW

To understand the desired functionality of the CHARm WebIF, one must understand the system requirements, i.e., to identify who needs to view child-health information, to understand why they need it, and to understand the various procedures that need to be followed when accessing such confidential information. The best way to visualize a software system is to first identify different objects associated with the system and then seek to understand how they collaborate with each other to accomplish a task. Also, the tasks or activities of each object of a given system are the most crucial step in identifying the requirements of the system. In other words, identifying the objects and studying their relationships to each other is the key step in analyzing the system and its desired behavior. We follow such an object-oriented analysis approach for analyzing the system in question, given that this system’s basic functionality is directly related to analyzing the activities that a user will carry out for completing a set of tasks and to developing interfaces for the user to carry out those tasks efficiently. The analysis provides a sound basis for identifying the functionalities, actors, and actions associated with the system, making it simple to find out the desired behavior and capture meaningful requirements.

Further, an object-oriented approach helps in identifying the possible actors (objects) associated with the system, as well as identifying different activities associated with the actors, which in turn are the functionalities that are offered by the system. Often, the foundation of a design, as well as the development of said design, is to capture the
requirements and activities associated with different objects of the system, and this project has been no exception. Additionally, capturing the requirements of an object oriented system is best done using an object oriented language. We use *Unified Modeling Language* (UML) a general-purpose object-oriented modeling language to capture the system analysis, because it includes a complete set of graphical notation techniques to create visual models of the object-oriented software systems [1].

We analyze the system and revise the requirements based on that analysis again and again until the captured requirements were in agreement with the stakeholders. Each review with the stakeholders provides new functionalities and features for the system. Also, each revision of requirements helps in further identification of new functionalities, actors, and actions; and the revisions are completed after all the desired functionalities are identified and are in agreement with the stakeholders.

This chapter focuses on system analysis, which, as stated above, starts by identifying the requirements of the system and the functionality the system will offer. Also, it helps in identifying different actors associated with the system, user goals, and actors’ roles in the system, and the activities that the user carries out. The system requirements are systematically documented as the functional requirements and non-functional requirements listed in Sections 3.2 and 3.3 respectively.

Both the functional and non-functional requirements together form a strong basis for the testing. The class diagrams in Section 3.4 describe the next steps in the process, i.e., identifying the system’s structure in terms of key classes and the relationships they have with each other.
3.2 Functional Requirements

This section discusses the process of requirement collection. The most important aspect of requirement collection is to understand the needs of the CHARM WebIF and what it is trying to accomplish. A systematic analysis of the system resulted in the following functional requirements for CHARM WebIF. Each requirement lists an activity associated with the CHARM WebIF, and which included as a part of the services that the CHARM WebIF offers.

3.2.1 Registration

This feature allows a user to register with CHARM. Additionally, it provides security to the system by only allowing registered users to access the system.

3.2.2 User Roles

The CHARM WebIF allows the user to have different roles. Privileges are based on the user’s role(s). A particular role can have more privileges than another.

The current available roles are:

- Administrator
- Account manager
- NBS user
- UMD user

3.2.3 Record Personal Information

This feature records a person’s demographic information as well as other information like postal address, contact information, etc. The information that is recorded is as follows:

- Person’s name (first name and last name)
- Organization name
3.2.4 Login and Account Information

The CHARM WebIF records account information as well as other account security details. The information recorded is as follows:

- Login name
- Password
- Security question
- Security answer

3.2.5 Forgot Password

This feature allows a user to retrieve his/her password whenever necessary. The user must provide the username in order to request the password. The password request is acceptable only if the user provides a correct answer to the security question.

3.2.6 Edit/Update Information

Users and administrators may wish to change their information over time. Also, the administrator might need to change another user’s information over time. This feature helps the users accomplish this task.

3.2.7 Update User Status

The administrator might want to update a user’s status based on the user’s activity. The default user status is new user. The administrator can directly activate a new user account. Other available status options are:

- Active
- Inactive
3.2.8 Search User

The administrator might need to search users registered with CHARM. In order to do so, s/he might use the search user feature with the following options:

- Search by first name
- Search by last name
- Search by organization name
- Search by account type (active, inactive, new user, all accounts)

3.2.9 Search Child

To find a particular child in the CHARM CORE database, the user must provide the following information.

- Child’s general information:
  - First name
  - Last name
  - Middle name
  - Gender
  - Date of birth
  - Social security number
  - State file number
  - New born screening number

- Child’s birth information:
  - Birth weight
  - Birth order
  - Birth city
  - Birth state

- Child’s contact information:
  - Address line 1
  - Address line 2
  - City
  - State
• Zip code
  • Phone number

• Mother’s information:
  o First name
  o Last name
  o Middle name
  o Maiden name
  o Date of birth
  o Social security number

• Father’s information:
  o First name
  o Last name
  o Middle name
  o Date of birth
  o Social security number

• Legal guardian’s information
  o First name
  o Last name
  o Middle name
  o Date of birth
  o Social security number

### 3.2.10 Query for Child’s Health Information

A user can query on any child based on the query privileges the user has been given. User privileges are based on the user’s role. Possible roles are NBS user, UMD user, administrator, and account manager.

Based on the user’s role, the queries that can be requested are as follows:

• Administrator/NBS user queries:
  o Immunization queries
  o EHDI queries
  o Birth record Information queries
• NBS queries
  o BTOTS queries
  o Core queries
• UMD queries
  o Hearing screening results
  o Immunization history

3.3 Non-Functional Requirements

The CHARM WebIF is an addition to many existing components of CHARM. Consequently, it needs to coordinate its services with the other components. For this reason, the CHARM WebIF must comply with the development process and the standards that are used in CHARM. Following are the non-functional requirements for CHARM WebIF:

3.3.1 Development Environment

The development environment for the CHARM WebIF must be the same as for all other components in CHARM. The implementation constraints are as follows:

• **Operating system:** The CHARM WebIF can be developed in either a Windows- or Linux-based operating system.

• **Language and platform:** The CHARM WebIF must be implemented using Java-based technology, such as Java server pages and servlets. Detailed information can be found in Chapter 7.

• **Web server:** The CHARM WebIF requires an Apache server, which is an open source HTTP server for Linux and Windows.
• **Web Container:** The CHARM WebIF requires Apache Tomcat, which is an open source Web Container.

• **Database System:** The CHARM WebIF is intended for allowing access to child information to limited users with specific privileges. The user information along with the access specification is from the database, which uses a Postgres database system.

3.3.2 **Integrity Constraint**

The data changes and updates are made as a result of user interaction with the CHARM WebIF not the data itself, and as such the integrity of the data is maintained.

3.3.3 **Quality Control**

Thorough unit testing, usability testing, and integration testing are essential to ensure performance and quality. Thus, we conduct tests in these key areas.

3.3.4 **Documentation**

Our design document and user guides, as well as this report, assist users in understanding the design and different functionalities that are supported by the CHARM WebIF.

3.3.5 **Release Management**

Once the code is tested on the local environment, we push testing into the development environment and then to production.

3.4 **Context Analysis**
The next step after the system analysis and the requirements collection is to identify the context of this system, i.e., the surrounding components that use the system. In other words, the context diagram identifies the actors outside the system that can interact with the system. Such analysis is considered as the highest level view of the system because it shows the overall system interaction. Figure 3.1 illustrates the context diagram of the CHARM WebIF.

![Context Diagram](image)

**Figure 3.1: Context diagram for the CHARM WebIF.**

As shown in Figure 3.1, the CHARM WebIF primarily interacts with two other components, namely, the matcher and web-agents. The matcher is responsible for matching a person record in the CHARM CORE database. The CHARM CORE database contains a person’s demographic information. The CHARM WebIF communicates with the matcher in order to locate the child regarding which the user is seeking information. Beyond the matcher, the CHARM WebIF interacts with web-agents for querying health-
related data for the child. The web agent finds patient related information from different participating programs and sends it back to the CHARM WebIF.

3.5 User Goals

We capture user goals in order to identify all actors associated with the system and capture activities associated with the system’s functionality. The use case diagrams for the CHARM WebIF captures the goals of the CHARM WebIF end users, including the general user (a user that does not belong to any specific participating program), the administrator and other end users like a Utah Master Directory (UMD) user or a Utah Health Information Network (UHIN) user. Figure 3.2 shows the CHARM WebIF users, Figure 3.3 shows the general goal of a CHARM WebIF end user, Figure 3.3 illustrates the goals of users including a general user, UMD, and UHIN user, Figure 3.4 shows the goals of an administrator, and finally, Figure 3.5 extends the goal information.

![Use case showing the basic users of web interface](image)

**Figure 3.2: Use case showing the CHARM WebIF users.**
Figure 3.3: A use case showing the goals of the CHARM WebIF users.

Figure 3.4: A use case showing goals of a general user.
Figure 3.5: A use case showing goals of an administrator.

Figure 3.6: A use case showing the search user functionality.
As illustrated in Figure 3.4, any general user of the system has a goal of finding a child’s information. Each type of user enters the system differently according to their respective privileges, but they all achieve the same goal. Other users such as a UMD or a UHIN user are redirected to CHARM WebIF after the authentication by their respective participating programs; their end goal is the same as that of any CHARM user.

Figure 3.5 shows the goals of an administrator. This special user of the system is responsible for maintaining the other users of the system as well as assigning them privileges. The administrator can approve a user for accessing the CHARM WebIF, update or edit the user’s information, assign roles to user, search a user, and find child information.

Figure 3.6 illustrates the goals in further detail. This goal is associated with the administrator and aids the administrator in searching the current users of CHARM WebIF. The administrator can search the user with any one of the available search criteria. The search criterion is based on filtering the users from their first name, last name, organization name or their account type. The account type can be further divided in 4 sub-categories like new user, active user, in-active user or all users. An administrator can use any one of this filter to search the desired user.

3.6 Structural Analysis

The process of requirement collection helps in understanding the system and the key functionalities offered by the system. It also provides a sound basis for defining the structure of the key classes associated with the system. This section explores the key classes that resulted from analyzing the functional requirements. Figure 3.7 shows the
structural overview of the CHARM WebIF along with the key classes and the interactions between objects. A brief overview of the classes follows:

- **User**: A user class represents an authorized user having privileges to access the CHARM WebIF. This class contains basic demographic information.

- **Role**: A role class represents the role of a given user in the system. Current roles include roles for an administrator and other end-users. This class is responsible for providing users with privileges for accessing information.

- **Address**: This class represents the address information for a particular user of the system. This includes the street address, city, state and zip.

- **Contact Information**: This class represents contact information other than a mailing addresses, such as an email address and phone number.

- **User Account Information**: This class represents the user account information for any user of the system. This class maintains information regarding the unique username, password and other security information for each user.

- **Status**: This class represents the status of a given user in the system. Current status levels include new user, active user, and inactive user.

- **User Account**: This class contains all other classes and is the key class in the CHARM WebIF. The attributes are the combination of the attributes present in the contained classes.
Figure 3.7: Analysis of class structure for the CHARM WebIF.
CHAPTER 4
ARCHITECTURAL DESIGN

4.1 OVERVIEW

The architectural design of WebIF centers on a backend consisting of application layer classes that handle communications with CHARM’s query manager and access to a user-account database. Both the application layer classes and database schema are based on the result of the requirements analysis gathered as the first step of the development process. The application layer classes play an integral role by handling the business logic of the application and doing the necessary communications with the database layer for processing a user request. The application layer classes communicate with both the user interfaces and the database to maintain a smooth flow among the activities offered by this system.

CHARM WebIF’s design is object oriented. The application layer classes communicate with each other to accomplish user goals. Also, the interaction of user with the CHARM WebIF is captured using UML sequence diagrams that help in understanding this process. The database classes and relationships are captured using entity-relationship (ER) diagrams, which is a database modeling method. This method provides an abstract and conceptual representation of data to produce a conceptual schema of a system like a relational database and thus helps in better understanding of the relationships among different database tables.
4.2 User Interface Design

Figure 4.1 shows the complete flow and interaction of the different web pages associated with the CHARM WebIF.

As shown in the diagram, any user can register to CHARM by entering valid information in the “New user registration” page. A login page is available for all users to login to the CHARM WebIF. After successful login, the user is directed on the basis of his/her role. If the user’s role is that of an administrator, s/he will be directed to the “Admin home page,” while the home page for any other user is the “Search child” page. After successfully logging into the system, the general user can search for a child, query on the child, or change account settings. An administrator can do all of these tasks as well as approve any new user or change user account details.
Figure 4: A diagram showing web pages in Web Interface
4.3 Application Layer Design

This layer contains the classes that are responsible for rendering the data on web pages and retrieving requests from the user interface. Additionally, based on the request, this layer generates a response for the user. The description of the application classes in Figure 4.2 is as follows.

- **Read Properties:** This class is responsible for reading the information from the configuration file. The configuration file is an xml file and contains information related to the database, configurable parameters and other connection information (required to connect to other components).

- **Form Bean:** This class is responsible for reading the user input relayed from the registration form (form used by users to register for charm) and validate those inputs. This class is also responsible for inserting the user data into the database and triggers a message to the administrator for approving the newly registered user.

- **Register Mail Bean:** This class is responsible for sending out an email message to the administrator whenever a new user registers with CHARM. This class is dependent on the Form Bean class as the user registration process is carried out by that class.

- **Login:** This class is responsible for carrying out the user authentication process. It reads the user input from the login page, and based on the user input, checks the existence of the user in the database making sure only users with an active status are allowed to login to the system.
• **Username Validation:** This class comes into play when a user has problems in accessing his/her account. The most common reason can be a mismatch in password. The first step in getting the password for the account is to verify the user name. This class verifies the username thereby providing the security questions the user selected while registering for the account.

• **Answer Validation:** This class verifies the security answer for a particular user. This class is dependent upon the Username Validation class for the security question for a particular username. After the answer is verified, the user is sent the password by email.

• **Mailer Bean:** This class is responsible for sending out the email to the user if the user provides the right answer to the security question. This class is dependent on the Answer Validation class for checking whether the user has provided the right answer.

• **Search Result:** This class is responsible for carrying out the search functionality when the administrator provides any details, such as, first name, last name, and organization name or account type. This class queries the database according to the search requirements and returns the result.
4.4 Database Design

This section describes the design of the database used for this project. An ER diagram [5] is used in order to show the database structure. The database tables are shown in Figure 4.3 and are discussed below.

- **User Account:** This table contains all data related to the user. The data in this table includes first name, last name, title, organization name, complete address (street name, city, state and zip), contact information (phone number, email), login information (login id, login password), security information (security question, security answer), status (local to the system) and role name (local to system). The primary key for this table is loginID, while status and roleName are
Figure 4.3: ER diagram showing database design for the CHARM WebIF,

foreign keys corresponding to the StatusCode and RoleCode tables, respectively. This table is used to identify a user, its role, and its current status.

- **Role Code**: This table contains data related to the role of any given user in the system. Along with the role name, a description of the role is also included. This table is used for adding or deleting roles. The role of the user is important as it can decide the privileges the user can have.

- **Status Code**: This table contains data related to the status of the user. Along with the status, a description of the status is also included. This table is used for adding and deleting user status. The status of a user is important as it allows or denies a user access to the system.

- **Query Names**: This table contains data related to the queries that are used to extract information about a person in CHARM. This table is primarily used for displaying information on the JSP page. As in the case of a query header, query
name and query value information is displayed on the page. The primary key for this table is the query display name that is the used to display the query name on the page.

- **User Query**: This table contains data related to the user’s role and the query names that the user has privilege to access. This table is also used for displaying accurate information to the user based on his/her role in the system. This table does not have any primary key and is related to the Role Code and Query Names table by a foreign key relation. This table is important as it provides access rights to queries. The role specific query information is extracted from this table.

**4.5 System Interaction**

This section describes the interaction diagram for the CHARM WebIF. The interaction diagram provides useful information regarding the order of the processes executed in order to accomplish a task. Such diagrams give an overview of how the processes operate with other processes as well their order of occurrence. Also, these diagrams help in analyzing the system’s behavior.

Figure 4.4 is a sequence diagram showing the administrator’s interaction with the system. The administrator can login to the CHARM WebIF by providing the login credentials like user name and password. If correct, the administrator can approve any new user by assigning him/her a role from the list of available roles. The administrator can assign a role and status to other users. The administrator can also search a user on the basis of first name, last name, organization name, and account type. Finally, the administrator can view, edit, or update user account details.
Figure 4.4: Sequence diagram showing the administrator’s interaction with the system.

Figure 4.5 is a sequence diagram showing the interaction of a general user of CHARM WebIF. A general user can edit/update their account details, search child, view child’s information, and query child for other health related information. All tasks can be performed only after successful login to the system.
There are other possible users who can directly access the CHARM WebIF directly without going through the login page. Currently, these user groups are limited to UMD and UHIN users. More user groups can be added to the list as needed. Figure 4.6 is a sequence diagram showing the UMD user group access to the CHARM WebIF.

Figure 4.5: Sequence diagram showing a general user’s interaction with system.
Figure 4.6: Sequence diagram showing a UMD user’s interaction with the system.

A UMD user follows an external link to get to the CHARM WebIF. Once the user has reaches the UMD gateway page, the UMD web service queries for the user’s credentials. After the credentials are received from the web service, a query to the CHARM WebIF database is done for checking if the user already exists. If the user exists, he/she is redirected to the child search page, otherwise to the no authorization page.
CHAPTER 5
USER INTERFACE DESIGN

5.1 OVERVIEW

User interface design is an integral part of CHARM WebIF as it allows the users to interact with the CHARM WebIF to accomplish some task. To facilitate this process, it is extremely important to design user interfaces that are task oriented and help in efficient retrieval of information. Making the interfaces task oriented requires the designed understand user needs, as well as the user’s primary purpose of accessing the interface. The requirement definition is used as an aid to create the interfaces, as the requirement definition captures all the features that the CHARM WebIF offers. The rest of this chapter is dedicated to understand the user interface in general and understanding the design choices that were made for each interface of the CHARM WebIF.

5.2 User Interfaces

Designing user interfaces is an extremely challenging task. The user interface plays a crucial role in directing the user through a step-by-step procedure to achieve his/her ultimate goal. User interfaces are designed specifically for easy use, for making the user’s task of learning entire process more streamlined. An important key in designing a user interface is to understand the user requirements. Once the user requirements are known, the content can be arranged for ease of use while providing all the salient features.
5.3 Design and Organization of a User Interface

The overall design of the user interfaces for the CHARM WebIF rests on user behavioral patterns. The content on the interface is laid out following the most popular patterns used for dividing the content of the web pages. The physical structure for CHARM WebIF is one-window paging [6]. Other factors considered in designing the layouts are gestalt principles [6]. This section explores all the patterns used in each interface along with their importance.

5.3.1. Login

This web page allows the user to login to the CHARM WebIF and get access to all the facilities that are offered by the CHARM WebIF. Figure 5.1 shows the screenshot for the web page.

![Login to your CHARM Account](image)

Figure 5.1: A screen shot of the login page.
The content for this page is very limited and is laid out in a straightforward way such that the user easily knows what is expected and can easily navigate. In other words, the interface is user friendly. As seen in this interface, the pattern used here addresses the issue of navigation, in particular. The structure of this application is closely tied to navigation between pages. This pattern is called clear entry points [6]. This pattern restricts access to navigation, thereby providing clarity in understanding the application. The interface has only a few entry points; therefore, it has clearly defined tasks that the user can do. This allows the user to have clear guidance on what to do next. If the user can understand the interface and find it easy to get started for any of the tasks, s/he can achieve instant success [6]. There are also some other navigation links in this page like the UDOH homepage and a “contact us” link. These links are not a primary source of interaction for the user and hence are barely visible to user. In terms of grouping and alignment, the login page follows gestalt principles [6], including proximity, similarity, continuity, and closure. The CHARM logo is the focal point of the page. There is a nice visual flow that starts from the CHARM logo, follows down to the text fields, and finally comes to the login button. Another noticeable alignment is the left-right alignment between the labels and the text fields. The left-to-right alignment makes the user identify the connection between the labels and the text fields next to them. Also, the page has a liquid layout, which helps user work in less space.

5.3.2. New User Registration

This page allows the user to register for the CHARM WebIF. Once the user provides all the required information, a validation check is performed on the information
provided by the user. After the information is validated, the system sends a notification to the administrator regarding the new registration. The screenshot for this page is shown in Figure 5.2.

![Register for CHARM Account](image)

**Figure 5.2:** A screenshot of the user registration page.

The user interface for this page is organized such that it follows the same visual framework [6] as the other pages in the CHARM WebIF. This makes the user interface consistent and user friendly. Similarity among the layouts helps the user feel comfortable
with navigation. Another pattern that can be seen here is the left-right alignment between the labels and the text fields. In terms of grouping and alignment, then, the registration page follows gestalt principles [6]. Here, we can see proximity, similarity, continuity, and closure. Again, the CHARM logo is the focal point of the page. There is a nice visual flow that starts from the CHARM logo, follows down to the titled sections, and finally comes to the buttons (reset and create account). The titled section defines separate sections of content by giving each of them a visually strong title that relates to the content of that section. This makes it is easier to scan the information present on the page, thereby making the information architecture obvious to the user.

Another pattern that can be seen here is diagonal balance [6]. The page elements are laid in a way that puts the visual weight to both upper-left and lower-right corners. Other than merely enhancing the look of the page, the diagonal balance sets up the page in a way that the user’s eyes moves easily from top left to bottom right. Also, the page has a liquid layout [6], which helps the user work in less space, thereby shrinking the page as per a user’s requirement. Another important navigation pattern seen in this page is the escape hatch [6]. The “Already Registered” tab at the leftmost top of the page allows the user to return to the login page in case s/he already has an account and has accidently navigated to this page. This helps the user to get back to a known place and not get trapped in an unwanted and unknown page, hence making navigation easier.

**5.3.3 Forgot Password Page**

This page allows the user to retrieve a lost password. This page plays a crucial role when a user somehow forgets the password and needs to login to the CHARM
WebIF. The user simply clicks on a link called “forgot password” on the login page and is directed to this page. The initial requirement in order to retrieve a password is to provide a valid username. Only when the user provides the valid username is s/he directed to another page showing the user the security question said user created at the time of registration. The password is emailed to the user only if the user provides the correct answer to the security question. Figure 5.3 shows a screenshot for this initial page wherein the user is asked for the username. Figure 5.4 shows the screenshot of the recover password page in which the user is asked the security question.

Figure 5.3: A screenshot of the forgot password page.
Figure 5.4: A screenshot of the recover password page.

The design of these two pages is purposefully very much alike. Additionally, both pages follow the same visual framework [6] as the other pages in the CHARM WebIF. We can see here is the left-to-right alignment between the labels and the text fields. Again, in terms of grouping and alignment, the registration page follows gestalt principles [6], namely, proximity, similarity, continuity, and closure. The CHARM logo is the focal point [6] of the page. There is a nice visual flow that starts from the CHARM logo, follows down to the text, and finally comes to the submit button. This visual flow also part and parcel of the diagonal balance [6]. An escape hatch [6] “Homepage” can be seen in both pages in the left most corners.

5.3.4. Administrator Home Page

This page is the home page for the administrator. Hence, only users with an administrator role are able to access this page. This page helps the administrator to organize the users by approving them for accessing the CHARM WebIF, view/update user information, search users, or change his account settings. Figure 5.5 shows a
screenshot for this page. As we can see in the figure, there is a list of new users that have recently registered with the CHARM WebIF. The admin can approve each user by clicking on the “Approve” button, which then brings up a page showing the detailed user information. The administrator can also search a user by providing any one of the search criteria, such as, first name, last name, organization name, or account type. Other links on this page are “Settings” used for changing the account settings for administrator, “Child Search” used for finding a child in the CHARM database.

![Image of admin home page]

Figure 5.5: A screenshot of the admin home page.

As to the design of this page, we can see that this page follows the visual framework of the other pages [6]. Also, note the left-to-right alignment [6] of search criteria wherein the text labels are followed by the text boxes.
In terms of grouping and alignment, the registration page also follows gestalt principles [6]. Again, we see proximity, similarity, continuity, and closure, the CHARM logo is the focal point of the page. Here, a lot of information needs to be presented to the user. The challenge is to present the data in a way that the user can easily find the piece of information s/he is looking for. In this instance, the administrator is looking for all the new users who have recently registered for an account. Therefore, we used tabular representation of data with row striping [6]. The data in the rows represent the information of the registered user as a whole. This also helps the user to read row-wise and not column-wise. Another feature is the use of section titles in the table to clearly delineate headings for the content in the columns. The caption of each table is placed close to the tables, following the gestalt principle of proximity. Another navigation pattern that is seen here is global navigation [6]. This pattern helps the user to navigate through the CHARM WebIF. Also, it makes the UI structure obvious to the user. Any section that the user wants to explore is just a click away from user. The global navigation features occupy very little space on every page and show a consistent set of links/buttons/tabs to provide access to key sections of the entire application. In this page, the global navigation bar consists of links to home, settings, child search, and logout. The header and footer, and the color scheme help in creating a visual framework for the entire application. This visual flow also leads to the diagonal balance [6]. An escape hatch [6] “logout” helps the user return to the initial page at any time. Finally, the page has a liquid layout [6], which helps the user work even in less space thereby shrinking the page as per one of the captured user’s requirements.
5.3.5. Account Settings Page

This page is used to change the current settings of an account. Any user of the system can use this page. To do so, the user clicks on the settings tab in the global navigation bar in order to access account details. Before showing the user the account details, however, password verification is performed. After the password is verified, the user is directed to the account details page wherein s/he can either view or edit the account or personal information. Figure 5.6 shows a screenshot of this page.

![Account Information](image)

Figure 5.6: A screenshot of the account details page.
The header, footer, CHARM logo, and the color scheme create a visual framework [6] like the other pages in the CHARM WebIF. This makes the user interface consistent and make it user friendly. The similarity between the layouts helps the user feel comfortable with navigation. Another pattern that can see here is the left to right alignment between the labels and the text fields. In terms of grouping and alignment, the account details page follows gestalt principles as well [6]. Here, we see proximity, similarity, continuity, and closure. The CHARM logo is the focal point of the page. There is a nice visual flow that starts from the CHARM logo, follows down to the titled sections, and finally comes to the buttons (reset and update). The titled section defines separate sections of content by giving each of them a visually strong title, which relates to the content of that section. Such page layout makes it easier to scan the information present on the page, thereby making the information architecture obvious to the user.

Another pattern that can be seen here is the diagonal balance [6]. The page elements are laid in a way that puts visual weight to both upper-left and lower-right corners. Other than enhancing the look of the page, the diagonal balance sets up the page in a way that the user eyes moves easily from top left to bottom right. Also, the page has a liquid layout, which allows the user to work even in less space, thereby shrinking the page as per a captured user requirement. In this page, the global navigation bar consists of links for home, settings, child search, and logout. An escape hatch [6] “logout” helps the user return to the initial page at any time.

5.3.6 Search Child Page
This page is one of the key pages of this application. All users of the system irrespective of their role access this page. This page facilitates the child search and has a very goal-oriented interface. Figure 5.7 shows a screenshot of this page. This page is similar to the other pages in the CHARM WebIF and hence follows the same visual framework. The information on the page is laid out to create a proper diagonal balance.

The global navigation features facilitate navigating through different pages of the CHARM WebIF. Also, it makes the UI structure obvious to the user. Any section that the user wants to explore is just a click away. In terms of grouping and alignment, this
Figure 5.7: A screenshot of the child search page.

page follows the gestalt principles [6] of proximity, similarity, continuity, and closure. Again, the CHARM logo is the focal point of the page. The titled section defines separate sections of content by giving each of them a visually strong title that relates to the content of that section. This way, it is easier to scan the information on the page, thereby making the information architecture obvious to the user. Another pattern unique to this page is closable panels. Since there is too much information to be displayed on this page, closable panels allow the user to access all information associated with this page without having a cluttered screen.

Therefore, the information is divided into clearly named sections. The user can choose to view one or more at a time. The structure is flexible; not having a fixed size for each panel saves a lot of space. Also, the page has a liquid layout, which helps the user work even in less space, as per a captured user requirement. An escape hatch “logout” helps the user return to the initial page at any time.

5.3.7. Child Query Page

On this page, child information is shown on the left and the queries that can be done for that particular child are shown at the right. This page is similar to the other pages and therefore follows the same visual framework. A global navigation facilitates the navigation from one page to another at any time. Titled sections make it easier to read separates child information. Similarly, queries are chunked based on titles, which is another form of section titles. The closable panel allows the user to look at only the required information. There is a smooth visual flow from left to right also creating the
right diagonal balance. In terms of grouping and alignment, this page follows gestalt principles [6].

Figure 5.8: A screenshot of the query page.

5.3.8 Get Query Results

This page shows the results of a query requested on a particular child. Results are clearly displayed using section titles as visual guides. Each query is a title and therefore it
is more readable and clear to understand. A global navigation bar allows the user to navigate to other pages. Also, global navigation here allows the user to start a new search by going back to the child search page. The information is shown in a tabular structure, which makes it easier for the user to read and understand. Figure 5.9 shows the screenshot for this page.

Figure 5.9: A screenshot of the query results page.
CHAPTER 6
USER GUIDES

6.1 OVERVIEW

The user guide presented in this chapter provides an instructions for using the CHARM WebIF for both end-user and administrators tasks. Also, the user guide discusses the functionalities that each user have in additional to the general functionality.

6.2 General User Guide

The general user guide is for any user who is not an administrator. Current general users of the systems are NBS user, UHIN user and UMD user. General users have privileges associated only with their own accounts. They can only change or update their own account but are not allowed to view or change accounts of other users. Other than that, a general user is allowed to search a child in the CHARM database but has only limited access to queries that they can request on a child.

This section presents each section of the general user guide along with screenshots. Each screenshot is followed with the usage description and other related information.

6.2.1 New User Registration Page

This page carries out the user registration process. The user needs to provide all the mandatory information, in order to register for a CHARM account. The information provided by the user is verified, and the user is kept on a hold until the administrator approves the user account. The user must provide personal information as well account
information. The account information is used to login to the CHARM WebIF. The user must comply with the requirements of registration which includes selecting a valid password (a link for password validation rules can be found next to the password textbox) as well as agreeing to the confidentiality policies. Figure 5.2 shows a screenshot of the new user registration page.

6.2.2 Login

This is the first step of the user’s interaction with the CHARM WebIF. Every user needs to login using this page in order to access the other features of the CHARM WebIF. Other users who are redirected to the CHARM WebIF from any other source are generally using the CHARM WebIF for accessing the child’s information. In such cases, the user is already authenticated in the prior source and does not need to login again for accessing child’s information.

Figure 5.1 shows a screenshot of the login page. The user must provide the login credentials in order to login to the CHARM WebIF. Once the user login credentials are verified, the user can access other features of the CHARM WebIF. Only users that are registered with CHARM are able to login with this interface. The other two links present on this page are “I forgot my password” which is used in case of a lost password and “Request an account” used for requesting an account for registering with CHARM.

6.2.3 Forgot Password

This page helps a user in retrieving a lost password. A link for this page can be found on login page. The primary requirement for requesting a lost password is to verify your username. Figure 5.3 shows the initial screen that pops up for retrieving the password.
Only if the username is verified, the security question for the username is retrieved. Figure 5.4 shows the screenshot of the next screen that is shown to the user if the user enters a valid username. This screen shows the username along with the security question selected by the user at the time of registration. The user must provide the right answer for the security question in order to retrieve the password. Once the user provides the right answer, an email is sent to the user on the email address provided at the time of registration. The email contains the user’s password which can then be used to login to the CHARM WebIF.

**6.2.4 Enter Child Details Page**

After the user enters the username and password on the login page, s/he is redirected to this page, which acts as the homepage for all general users. Any user who is not an administrator of the system will be directed to this page. This page allows the user to search a child by providing the information of the child. The search criteria for a given child are specified on the page and will work with only valid information. For instance, one must provide an accurate birth date and in the required format. No fields are mandatory for this search. But, for a successful search, the user must provide some valid information. Other than searching the child, the user can also change his/her account settings by clicking on the account setting present on the navigation bar. Figure 5.7 shows the screenshot for the same.

**6.2.5 Ask For Queries**

This page displays the information of the searched child from the CHARM’S CORE database. The information for the user is displayed along with a list of queries the
user is allowed to do on this child. The list of the available queries depends on the user’s role in the system. For general users, the list is limited to only few queries. Figure 5.8 shows the screenshot of this page.

6.2.6 Get Query Results Page

This page shows the results for the queries listed in the “Ask for Queries” page. This page shows demographics and alerts in addition to the other requested information. Figure 5.9 shows the screenshot for the same. The page has other navigation like starting a search all again (search child), change account settings (settings) and return to home page. A print button creates a printer friendly version of the page.

6.2.7 Account Settings Page

This page allows the user to change any information the user wish to change. The primary requirement for doing so is to verify your current password. Figure 6.1 shows the screenshot for this page.

![Verify Password](image)

*Figure 6.1: A screenshot of the password verification page.*
After the password has been verified, the user is redirected to the next page. Called the edit information page, here the user can see all the account details. Figure 6.2 shows a screenshot of this page. This page shows personal information, as well as account information. The user can change any information s/he wishes to change, except the login name. The login name is allotted at the time of user registration and must remain the same. A validation is performed on all the information provided by the user before it is updated. After the successful update of the user information, the user must

Figure 6.2: A screenshot of the edit information page.
remember the changes. For instance, if the user changes the password, s/he must remember on the change made in order to login in the future.

6.3 Administrator User Guide

The administrator user guide is for users who have been assigned the role of administrator. The administrator has all the privileges the general user has and also has some additional privileges.

6.3.1 Login Page

This is the first step in accessing the CHARM WebIF. Like all other users, the administrator needs to login using the login credentials. Figure 5.1 shows a screenshot of login page, which is same for all users.

6.3.2 Administrator Home Page

This page is the home page for the administrator. After the user logs in as an administrator, s/he is directed to this page. This page shows a list of the newly registered users and the search box. The administrator can either search any user or approve the user. If no users are awaiting approval, a message indicating such is shown along with the search box. The administrator can click on the approve button in order to view the user information and approve the user. A screenshot for the same is shown in Figure 5.5. If the administrator wishes to search any user, s/he can do so by entering information, such as first name, last name, organization name or account type in the search box. On the basis of the search criteria, a list of users is returned to the administrator. The user can the view the information of a given user by clicking on the view button placed against each user. A screenshot for the same can be seen in Figure 6.3
6.3.3 User Approval/ View Details

When an administrator wants to approve a new user or look up a current user, the administrator needs to view the information associated with the person. An administrator is the only person who can alter details about any user of the system. This action can be a result of an information conflict of some sort or simply general system maintenance.

Figure 6.4 shows the screenshot updating information on any user. The administrator can also change the role of a user. The current role of a user is displayed on the screen along with the current status. The administrator can change the user status to either active or inactive. The administrator cannot change the login id, first name, or the last name of the user, however. Possible actions are canceling the changes or save changes.
Figure 6.4: A screenshot of the view user information page.

Figure 6.5 shows the user information that is displayed to the administrator when s/he chooses to approve a user. By default, the user role is NBS query; the administrator can assign a different role or can let it be the default. Also, the status while approving a user must be active.

Again, the administrator cannot change the first name, last name, or login id for the user. The actions possible here are cancel the approval or activate the account.
6.3.4 Enter Child Details Page

The administrator can also search a child’s information by selecting child search on the navigation bar. This page allows the user to search a child by providing information of the child. The search criteria for a given child are specified on the page and will work with only valid information. For instance, one must provide an accurate birth date and in the required format. While no fields are mandatory for this search, for a successful search the user must provide some valid information. Other than searching the child, the user can also change his/her account settings by clicking on the account setting present on the navigation bar. Figure 5.7 shows the screenshot for the same.
6.3.5 Ask For Queries

This page displays the information of the searched child from the CHARM’S CORE database. The information of the child is displayed along with a list of queries the administrator can do on this child. Figure 5.8 shows the screenshot for the queries available to an administrator.

6.3.6 Get Query Results Page

This page shows the query result for the queries listed in the Ask for Queries page. This page shows demographics and alerts, in addition to the other requested information. Figure 5.9 shows the screenshot for the same. The page has other navigation tools like starting a search over again (search child), changing account settings (settings), and returning to home page (Admin Home Page). A print button creates a printer friendly version of the page.

6.3.7 Account Settings Page

This page allows the user to change any information the administrator wishes to change in his/her account. The primary requirement for doing so is to verify the current password. Figure 5.6 shows the screenshot for this page.

After the password has been verified, the user is redirected to the next page wherein all the account details are displayed; this page is called edit information. Figure 6.1 shows the screenshot for this page. This page shows personal information as well as account information. The administrator can change any information he wishes to change except the login name. A validation is performed on all the information provided by the user before it is updated.
CHAPTER 7
UNDERLYING TECHNOLOGY AND IMPLEMENTATION

7.1 OVERVIEW

The CHARM WebIF adopts the Model View Controller (MVC) design pattern, which includes a model component, view components, and controller components [12]. For the WebIF, the model component comprises application classes that hold the business logic of the application. These classes are accessed by the Controller which in this component is a servlet. The processed results are then directed to the view which is rendered by the JSP (Java Server Pages). The MVC design patterns helps in achieving one of the most important characteristic of OOP, i.e., separation of concerns. By using this MVC design pattern, the presentation (View rendered by JSP) is separated by the business logic (servlets and Java classes).

The project mainly uses Java and J2EE technologies like JSP and servlets. Also, since a major part of this project involves user interface development, it uses technologies like HTML, CSS, and JavaScript. The Java web technologies used for this project are a part of the Java EE platform along with the Java SE classes and packages. This chapter starts with a discussion of J2EE architecture, which is also the basis of this project. It next discusses the different technologies (JSP, Servlets, etc.) used in this project. Finally, this chapter discusses the different implementation challenges faced during the development of this application.
J2EE Architecture

J2EE provides a standard for developing multi-tier, enterprise services. It supports component-based development of multi-tier enterprise applications. J2EE applications are comprised of components (web components and EJB components), containers, and services. Web components are servelets and JSPs, while the EJB components contain server side business logic for the enterprise applications.

A J2EE application includes a client tier, middle tier, and enterprise data tier. In a client tier, generally web components like servelets and JSP are present. In the middle tier, enterprise beans and web services encapsulated the reusable business logic. The middle tier is comprised of two layers: a presentation layer and a business logic layer. Finally, in the enterprise data tier, the data is stored and persisted in a relational database.

7.2 Java Server Pages Technology

The JSP technology allows creating dynamic web content that is server and platform independent. A JSP page contains static data (represented in text-based format/HTML/XML) and JSP technology elements, which allow the page to construct the dynamic content. This page can also contain Java technology code for creating and processing the dynamic content of the page.

JSP separates the user interfaces from content generation. This allows creating flexible code that can be easily updated and reused. The JSP tags invoke the Java beans independently and are therefore a convenient way to create web applications that connect to server side Java components. The separation of HTML presentation code and the application logic make JSP most suitable for creating web pages.
7.3 JavaScript

JavaScript is a dynamic scripting language that allows prototype-based object construction. JavaScript functions both as a procedural language as well as an object oriented language. JavaScript is primarily implemented client-side to provide dynamic websites. The dynamic capabilities include runtime object construction, variable parameter lists, function variables, and dynamic script creation.

7.4 JavaMail API and JavaBeans Activation Framework

The JavaMail API provides a platform independent and protocol independent framework to build mail and messaging applications. It defines classes such as Message, Store, and Transport. The API can be extended to add functionality. The Java mail architectural components are the abstract layer (declares classes, interfaces and abstract methods for mail handling functions), internet implementation layer (implements part of abstract layer using RFC822 and MIME internet standards), and Java beans activation framework (to encapsulate message data and handle commands for interaction with data). Java beans activation framework is used by the JavaMail API in order to manage the MIME data. It helps in instantiating the appropriate bean for determining the type of a data, encapsulate access to it, and discover the options that are available on the data.

7.5 Java API for XML Processing (JAXP)

The JAXP is for processing the XML data using applications written in the Java programming language. It provides the capability of validating and parsing the XML documents. The most popular parsing interfaces provided by JAXP are Document Object
Model (DOM) and Simple API for XML parsing (SAX) interface. In addition to this, the API provides an XSLT (EXtensible Style sheet Language Transformation) interface for data and structural transformations in an XML document.

The DOM interface parses the entire XML document and constructs an in-memory representation of the document.

The SAX interface parses the entire document but does not create an in-memory representation. Instead, it informs the client of the XML document structure by invoking callbacks (methods).

7.6 JDBC API

The JDBC API is the industry standard for database independent connectivity between the Java programming language and databases. It is the core of the Java 2 platform. Also, the JDBC API is included in both the standard edition (J2SE) and the enterprise edition (J2EE) and provides server-side functionality. The JDBC API helps in establishing a connection with the database, sending SQL statements, and processing the results.

7.7 SOAP

(Simple object access protocol (SOAP) is a protocol for exchanging structured information. It relies on XML for its message format and on an application layer protocol like remote procedure call (RPC) and HTTP for messaging negotiation and transmission. SOAP is composed of an envelope (helps in identifying an XML document as a SOAP message), a header element (contains the header information), a body element (contains
call and response information), and a fault element (containing status and error information). SOAP is a protocol for accessing web service. It is primarily used in exchanging structured information in the implementation of web services.

7.8 JAVA Servlet API

The servlet API enables using the Java server technology in the network applications. Servelets are protocol and platform independent server side components. They provide a general framework for services using the request response paradigm. They provide secure web-access to data presented on the web pages. The Java servlet API is a standard Java extension API.

7.9 Apache Tomcat Server

Apache tomcat is an open source servlet container developed by the Apache software Foundation (ASF). It provides an HTTP web server environment for the Java code and implements the Java servlet and the Java server pages (JSP). The tomcat server includes tools for configuration and management, but it can also be configured by editing the XML configuration files.

7.10 Implementation Issues

We faced quite a few interesting implementation challenges during the development of the CHARM WebIF. They are listed as follows:

1. The primary use of the CHARM WebIF was in displaying the data to the users; therefore, the most challenging task was in organizing the data using renowned patterns. These patterns are concerned with the user’s behavior and comfort while
using any general interface. By using these patterns, the interface generated is more user friendly. The straightforward layouts make it easier for users to focus on their primary task rather than understanding how to use the interface.

2. Configuring the CHARM WebIF to make it portable to different environments was another hurdle. The configuration, like the database and connectivity to other components, was dependent on the testing environment which changed constantly for each test. Making the CHARM WebIF configurable such that it could be used across different testing environments was extremely important and was handled using a separate XML based configuration file also called deployment descriptor. DOM parsing technology is used read the configuration file and provided the required information to the Java classes. Also, the application-wide initial parameters were specified in the configuration file and read using a servlet context interface for the JSPs.

3. Another issue was the configuration of other involved components necessary for the proper functioning of the CHARM WebIF. The components that are required for the CHARM WebIF are the matcher and different web agents. Configuring each of these components as per the different development environments was challenging.

4. Integration of the CHARM WebIF with another existing web service was another implementation challenge. The CHARM WebIF needs to communicate with an existing web service in order to get the information of the user who is trying to access the CHARM WebIF. These programs are outside the scope of the CHARM WebIF; hence, communication and integration presented interesting challenges.
5. Since the CHARM WebIF is a web based component, session maintenance was another major issue. The session must be destroyed on an action like logout; otherwise, security is compromised.

6. Validation of user data was another challenge, as acceptance of any invalid data entered by user acts as a security hole of the system.
CHAPTER 8
SOFTWARE TESTING

8.1 OVERVIEW

Software testing [7] is an essential part of software development because it ensures quality as well as proper functioning of the product being developed. To test the performance and functioning of the CHARM WebIF, we perform unit testing, integration testing, system testing, and usability testing. This chapter discusses these testing techniques in more detail.

8.2 Unit Testing

Unit tests [8] are extremely useful in testing individual units of source code. The test determines if these units are capable of the proper functionality that is desired from them. The units are tested with a combination of inputs that include valid as well as invalid inputs. Unit testing helps in detecting errors in the development phase, as well as helps to rectify errors early in the software development cycle.

In the CHARM WebIF, each page is like a unit that needs to be tested for sequences of good and bad user input. We conducted different tests to assay the proper functioning of the CHARM WebIF, such as, entering an invalid value into fields, not entering any value in the mandatory fields, acceptance of valid values, etc.

8.3 Integration Testing

This testing technique is used to test individual software modules as a group. Different modules are integrated and are tested against their desired behavior [9]. In other
words, it is the combined testing of different parts of an application to determine if they
function together as desired and produce accurate results or if they have some troubles in
functioning as a combination. Such testing helps in identifying the problems between
different interacting modules that need to be integrated in order to provide the desired
functionality to the user.

All problems related to the interacting components are discovered in integration
testing and resolved before system testing takes place. For this project, we performed
integration testing with a matcher, as well as with different agents including the web
agent and the agents associated with different participating programs. Testing each
associated component with the CHARM WebIF ensured proper communication between
the components, as well verified that the communication resulted in the proper
functioning of the components.

As a result of integration testing, we were able to test whether the different
components communicate effectively and deliver their proper functionalities.

8.4 System Testing

This technique is used to test a system when it is integrated with other systems.
Since the CHARM WebIF is to be integrated with different other CHARM components,
it was extremely important to test CHARM WebIF’s integration with other software units
in CHARM. The purpose of system testing is detecting any inconsistencies between
different software units that are integrated together. System testing [10] allows testing all
components as individual components as well as a system. System testing helps in testing
the component based on the user requirement specification, system requirements, as well
as functional requirements. The complete system is configured in a controlled environment wherein each feature of the system is tested as it will be in a real environment.

System testing ensures the performance is as the system was intended, and that all the functionalities that are provided by the system are complete and accurate. It is based on overall requirement specifications.

The system testing for the CHARM WebIF was done by testing it with all the other components of the system as a whole. This required simulating the actual environment with all other actual components present as they would be in a real scenario. We performed testing on communication as well as overall functionality of the CHARM WebIF.

As a result of our system testing, we were able to verify that the component produces the desired result when it communicates with the other components of the system.

8.5 Usability Testing

This technique is used to test the product with different users. Usability testing helps in understanding how the user reacts to the product, as well as provides valuable information regarding the user’s goals and expectations for the product. This testing is extremely beneficial for any product as it helps in avoiding any future problems that might arise when real users begin utilizing the product.

As a result of usability testing for the CHARM WebIF, we made considerable improvements the user interfaces as per the user suggestions. Also, the design was altered
to make the interface more user friendly.
CHAPTER 9

SUMMARY

The CHARM WebIF is used to access the health data associated with a child within the overarching CHARM system. The CHARM WebIF integrates child information from different participating programs under a single interface, thereby providing a universal portal and single sign-on functionality to users. Only users who are registered with CHARM have access to the CHARM WebIF. Besides facilitating searches for a specific child and requests for health related information for a child, the interface provides other standard functionalities like managing account information for all users and other administrative functionalities, such as manage users, grant access rights to users, assign roles, and manage user information.

Currently, the CHARM WebIF’s application layer classes communicate directly to the database layer. Such access is not desirable as it is possible to compromise security at the database level. Therefore, a better approach is to introduce another service that is responsible for communicating with the database and carrying out all the required processing based on user request. The next version of CHARM WebIF will include a new authenticator service that eliminates direct communication between application layer and the database layer. Thus, the CHARM WebIF would no longer communicate with the application layer classes directly and exclusively use authenticator services instead.
REFERENCES


http://en.wikipedia.org/wiki/Functional_requirements


[6] User Interface Organization Principles and Patterns, Designing Interfaces by Jenifer Tidwell


[12] Model View Controller Design Pattern, Wikipedia,
http://en.wikipedia.org/wiki/Modelviewcontroller