Clowder:
A Software system to manage the accessibility of high performance computers for research

by

© *Samson Ugwuodo*

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of *science*

Department of *Scientific Computing*

Memorial University of Newfoundland

April 2017

St. John's Newfoundland

Abstract

Clowder is a system designed to help researchers in the Faculty of Engineering and Science to manage and control clusters of high performance computers. This system has existed for more than 3 years, but is yet to be completed because it lacks some features and functionality. Some of the features that are missing in this system is the ability to reserve a computer for a certain period of time, a function that allows users to add more computers and manage their properties , and also a user interface where users can have interactive access to the system. Therefore, there is a need to upgrade and complete this system by developing software that provides the missing functionality. A web interface addresses the issue of user accessibility. Having a dynamic database system provides the functionality of keeping record of user activities, storing computer information and managing all reservations. These design approaches are achieved using SQL models, HTML templates and the Go programming language. Instead of performing several command line prompt statements, which is the current method in the use of Clowder system, the software provides flexible user access to the cluster of computers, a dynamic system management in a single software system as a research tool in the Faculty of Engineering and Science.

Acknowledgements

I will like to express my humble thanks to my supervisor Dr. Jonathan Anderson for his overwhelming support, guidance, professional advise as a professor and his contribution through out this project.

I want to express my sincere appreciation to the School of Graduate Studies, the Department of Scientific Computing and Computer Engineering for their academical and financial support.

I would also like to thank the head of Department Dr Ronald Haynes , and Gail kenny for their subsequent advice and support through out my in class academic works and other wise.

Finally, I am grateful to my family and God almighty for the love and encouragement that i have been blessed with through out this journey.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The use of high performance computers for research in the Faculty of Engineering and Scientific has increase, because of increase in research tasks, such as testing new operating systems. As researchers demand large data storage and robust computer architectures, these computer systems and their properties also need to be expanded to accommodate this large tasks. As we expand the systems access and usage becomes more complicated to manage. Therefore we are faced with the issue of system management, system accessibility, and storage. These issues for example, are the inability to provide proper record of each computer system and their usage, the inability to provide flexible users access to the computers, and also the problem of monitoring the number of computers that are reserved by users. As a result of these challenges it is highly necessary to develop software that solves these problems for Faculty of Engineering and Science.

Clowder is a system designed to provide Preboot Execution Environment (PXE) for testing new operating systems through a Network File System (NFS) sever on

the high performance computers. Previous work has been started by researchers in Engineering department to complete these system for several years, but yet there was more improvement yet to be made in order to complete it, as it lacks some necessary modern functionality and features, such as database system, interactive user interface and restrictions. As an example, it takes extra effort and manual command line statements to access a computer in the cluster during research. Another reason for improvement is to provide flexible and dynamic user interface rather than current one (SSH). Therefore the current state of Clowder is not flexible and convenient enough. So the main goal of this project is to develop software that addresses the issue of user accessibility, system management and automatic control protocol. Addressing all these issues will improve the performance of Clowder in speed , access and robustness, and therefore making it a completed software tool.

We have accomplished this goal by using a design approach that provides a web interface, a databases system and automatic command protocols. The database system is designed to store and keep record of the computer systems and user activities. The web interface enable users to remotely log on to several machine from different locations simultaneously via a web page, and also provide user with the system inventory and other user activities. This software provide the ability to adding new computer system to the cluster, and to modify their properties individually. Also it provides the ability to make reservations: to allow users to reserve a computer for a certain period of time, and able to end reservations as well. A function to allow user search the inventory with some specifications according to their demands.

As this software serves as a tool to manage the cluster of computers and user activities, it is important to know that reservations made, computer details, network

interface cards and disks are stored as data. So all the computer system installed in the cluster with their names, vendors, memory size, architecture, and micro architecture is stored in the database. And same is applicable to the disks, network interface cards and any other devices that could be part of the cluster. Data is represented as variables in their various data types in the database scheme. All these information stored in the database tables servers as input data for the program and out put for the inventory. This database scheme allows user to add or update new machines installed in the cluster, and therefore provide dynamic access to the machines.

The web interface serves as a platform for users to interact with the system and overview other users activities via a web designated web page. This interface has also replaced the command line prompts which was the previous user interface for Clowder system. This choice provides flexible access and control to the system, by allowing users to log on to the system and make request of the inventory at any time through the web server.

Chapter 2

Design and Specifications

2.1 Specifications

The objective of this software is to design a dynamic user interface and a functional database system. With regard to that, the specification is made up of necessary features and functionality that appear on the user interface. The following are the major specification of this software:

- User Log in
- Data Inventory
- Searching Inventory(SI)
- Make Reservations
- Cancel Reservations
- Add data

- Update data

Registered users have access to the system with unique user name, so the user interface provide a log in form on the web page where user can submit their details to be stored in the database. The user interface is able to list the inventory of data stored in the database in difference categories. This is one of the main features on the user interface because it presents the content of the database. As there are bunch of data inventory on the user interface, it is necessary to have a function that allow users to search specify information from the inventory. Therefore the SI (search inventory) is a function designed to handle this issue. Another specification of this software is the ability to make reservations: this means that users are able to choose a machine from the inventory and reserve it for a fixed date and time. This function on the high level is meant to allow user to colonize a specific machine for running a particular task on that machine without interruption. When a reservation made is expired, another user will have the liberty to cancel that particular reservation to make the machine free. As new machines are added to the cluster, the Add-data function allow user to add details of machine to the database and update the inventory. Likewise, when user want to change a certain properties of the machines such as, memory size or disk, the Update-data function helps to perform this action.

2.2 Design

The design approach is based on the requirement to solve the issue of user interface, and system management. So we have designed this software with the intention to providing functional solution to the specifications. The design structure comprises

the various segments and element of the user interface (front end) combine with the database (back end) and the control protocols. We designed a web interface that provides flexible and dynamic access to the system with interactive functionality. This web page contains inputs and out put features that allow data to go in and out. The reason for choosing a web page instead of other options is to proving online dynamic and simultaneous access to the system rather than performing manual command prompts. Another reason is to have a functional real time management capability in the Clowder. The web page is managed by a web server, which serves a channel between the user and the database. We designed a operational database management system that address the issue of system management. The database stores data from the web page and also retrieve data to the inventory on the web page via the server. As user log on to the their account on the web page the server will pass this information to the database which is controlled by the main program (command protocol). The command protocols is responsible for handling the user interface activities and executing the database queries . Below is a figure that describe the main structure of the design in terms of operation.

Figure 2.1: The design structure

2.2.1 User Interface

The web interface represents the user interface (UI) where users get access to the system dynamically. It is the main part of the design because it represents the front end of the software.The web interface elements controls the basic functionality for data input and out put. It represent each feature as mentioned in the specification. This web interface structure is designed using HTML template as the front end and GO programming language as the back end. It contains all the necessary elements

required to have dynamic and flexible access to the system. These elements includes:

- Forms
- Input controls
- Navigational component
- Containers

Figure 2.2: User Interface Description

Forms

Forms are one the elements that provides the ability to input data to the database.The data is submitted through a HTML post request. The post forms are used for submitting data such as machines and reservations details. When this data is submitted with HTML form, the web server call the function handler to parse this data to the database. In this system design, the forms are used for creating new machines and reservation. The form has a text field for placing texts and submission button for initiating post request.

Input Controls

The input control in this web interface provides the necessary components that supports the functionality of the software. There is array of HTML input control, but we have chosen a few that satisfy our specifications. These input controls includes text field, buttons, date fields, drop down list and list box. They are used for both input and output functions like listing the inventory, inserting texts, viewing selection options and searching inventory. These controls are designed with HTML tags and adopted into Go functions with HTML template.

2.2.2 Server

The server is responsible for parsing all request from the web interface to the database and command protocol. When request is made on the web interface (for instance available machine on database), the server calls the specific function of command protocol to handle this request by providing the necessary resources. Likewise the

server interchange communication between the user interface and the database system for executions and requests.

Figure 2.3: Server description

2.2.3 Database

The database is another major component of this system, because it is provides storage and resource for the system. The data includes reservation records, machine details, disks, and NICs information. The database is designed with SQLite model

using Go programming language as the back end tech. SQlite was used here because its a fast open source SQL engine. We created a functional database system that suites the specification of the software. The database scheme contains tables of machines, user record, reservation, NICs (network interface cards) and disks. Each table is designed to have a primary key (PK) for unique identification. One of the function of the software is the ability to make reservations. So the reservation table has a foreign key (FK) reference to machine Id and user Id. This FK is used in reservation table to create a relationship between the reservation and users or machines tables. For example, when a User make a reservation of a particular machine, the program takes the Users' id (FK) and machine Id to create a reservation record in the database. Below show a typical model of the database design.

2.2.4 Command protocols

2.2.5 Design Data Flow

Figure 2.4: Database design

Chapter 3

Implementation

We have implemented this project in regard to the specifications. The entire system (clowder) is meant to be plugged to hardware and software base testing, but we have implemented the basic level of front-end (user interface) and back-end (database) protocols. Overall our approach have covered the user interface and database design with Go libraries and HTML templates.

3.1 Architectural Description

For a typical Go programming structure, our program depends on two main packages:

- DATABASE(db)
- HTTP(http)

In the db package contains all the .go file of database class (struct) for each hardware represented in the system. The http package also contain some .go file (for example

the Handler) and .html files of web interface content and elements. below is a table describing the list of header files in each packages.

Table 3.1: Database packages

No	Struct	Functions
1	machine()	initMachines()
		CreateMachine()
		UpdateMachine()
		GetMachine()
		GetMachines()

3.2 Code listing and Functions

Chapter 4

Evaluation

4.1 Test cases

4.1.1 Checking available machines

4.1.2 Reserving a machine

4.1.3 Updating a machine

4.1.4 filtering inventory

Chapter 5

Future Work

Chapter 6

Conclusion and Discussion

Appendix A

Appendix title

This is Appendix A.