

**Sri Lanka Institute of Information Technology**

‘Vigraha’ High-volume CDR Analyzing Framework

Title of Project: ‘Vigraha’ High-Volume CDR Analyzing Framework

Project ID:

Author: N.D.Dissanayake

Supervisor: Ms. Sujani S. Sumanadasa

Co-Supervisor:

Date of submission:

Abstract

This project will be targeted for telecommunication service providers to efficiently analyze CDR files in real-time and carry out advertising campaigns for target market segment. Analyzing CDR files in a normal system would cause the server to crash because of the amount of data. But our intention is to create a system that can work well under massive amount of data and still cause no harm to the server.

Introduction

Mobile service providers in addition to providing telecommunication services, provide external entities mass advertising campaigns. These campaigns need to be targeted to a specified market segment. To do this telecommunication service providers keep track of all their customers and their details by recording information of the customers’ calls and messages in a file called “***Caller Detail Record***” (**CDR**). These records hold information of the message or call. Normally when an external company asks for an advertising campaign the service provider will analyze the CDR files and determine which of the customers should be informed of the event. But the problem is the massive volume of files recorded; around 30 million records will be recorded within 24 hours, which makes it almost impossible for the service providers to analyze all the records. There are systems that handle a limited number of records but there is no system that can handle all the functions the service provider needs. “***Vigraha High-Volume CDR Analyzing Framework***” will address this problem.

So far no system has been able to achieve the full functionality of a CDR analyzer. Existing systems can only work with only one category of CDR, *for example voice calls or* SMS *but not both*. Additionally the current systems are very unreliable. They cannot handle too much data and have no way of handling crashes. This system will help telecommunication service providers the ability to carry out advertising campaigns easily and efficiently.

In order to handle the volume of data the system will have to be distributed. This will be our main priority, to handle the density of data. We will have to research on this to make it efficient and reliable. “***Apache Hadoop***” is the tool we are going to use for our distribution.

Objectives of the Project

The new system would truly revolutionize the mobile companies advertising, undoubtedly. And since this project is important we have found many objectives we must address. We have divided these objective 2 parts.

Main objectives:

* Create a system which will help mobile companies with advertising using CDRs.
* Create a system which can offer advertising to external companies.
* Handle the massive amount of data with limited server resources.

Secondary objectives:

* Inform mobile customers of events and offers given out by mobile companies as well as external companies.
* Select customers according to user defined categorizations.
* Handle unpredicted workloads.
* Create a system which can constantly run under high workloads.
* Invent a system to handle server crashes.
* Generate reports to required parties.

Research Questions

1. Will the server crash while operating, if so how can we handle it?
2. How can we handle so many records with so little resources?
3. What kind of categorizing will the companies require?
4. What kind of problems will occur with the system, since it is receiving so many data?
5. How can we distribute the system and still maintain its accuracy?

Methodology

The system’s main task is to analyze CDR files and select segment of customers, who will be informed of the event, offer or information that the company wishes to convey, by SMS.

When considering mobile service providers, presently, they offer many kinds of communication methods. For example SMS, Voice Calls, USSD, LBS. each of these leaves a CDR record behind with the service provider that holds information, like duration, recipient and location.

When a company needs to carry out an advertising campaign, say to people who are currently in the Colombo region, the above said company will first contact the mobile company requesting they do the advertising campaign. The company will take information on how they would want to segment the recipients, for the above example customers who are currently in Colombo. And the mobile company will input the system with the information. Each piece of information must be entered to the rule engine inform of a rule. These can take any form from Promotions, Loyalty programs or Tenure programs. And after that the system will start its analyzing process.

The system will first load all the information in the CDR files into the database and move the read files to a different place. This is done by the data loader. The data loader is event driven, and will periodically. If in a burst it finishes its tasks it will go to sleep. If it does not finish reading all the files it adds the leftovers to the next burst. This process should be running constantly. As the database fills up with data the rule engine can analyze them and choose candidates and send the message to them.

UI vise there are two types of users, Administrators and Report Generator. The Administrator level access allows users to define rules, create campaigns, edit them and create accounts, basically everything the system offers. The Report Generator users can only view reports which are generated by the system. The system creates summary reports that might be important to the mobile company and its external partners of the success of the campaign.

In addition to these tasks, we expect to add a feature to the system that will notify the systems Administrator of any unexpected crashes or events. This will help control the chaos it might cause in case of server faults.

Below is the high level diagram of the system.

Log Files

Admin - UI

Data Store

Reporting - UI

Rule Engine

SMSC

Hadoop Map Reduce

**Data Loader**

- Will load unread CDR records from files to DB.

- Read log files will be moved to a new location.

- Will be fired periodically.

Ex: Run each burst every 10 minutes. If all the records haven’t been read, add them to the next burst

**Figure 1**

My contribution

My responsibility is to create the Data Loader. The Data Loader is the starting node for system. From first time we install the system to the server up until the system is taken down it should be running. The Data Loader will be fired every once in a period of time, ideally every 10 minutes. Once it is fired it will start looking for CDR files, and each file it finds will be read and the information on it will be written to the database. And if it finishes reading all the record files before it’s burst time ends it will go to sleep. But the loader will not always be able to keep up with the incoming traffic, so most of the time it will leave files still waiting to be moved at the end of the burst. When this happens it will end that burst and start over as a new burst. This process must go on non-stop.

Our project requires log files or CDR files to perform. But in reality mobile companies are hesitant to give out information on their customers to external entities. So, because of this we have come up with a plan to simulate these files. We will use Perl for developing the simulators. I will be contributing to this as well.

**Data Loader**

- Will load unread CDR records from files to DB.

- Read log files will be moved to a new location.

- Will be fired periodically.

**Log Files**

- Record identifier

- Number Making the Call

- Number receiving the call

- When the call started (datetime)

- Duration

- Amount charged for call

- Result of the call (answer, busy)

- Call type

SMS

Voice Call

LBS

GPRS

USSD

Retrieve data

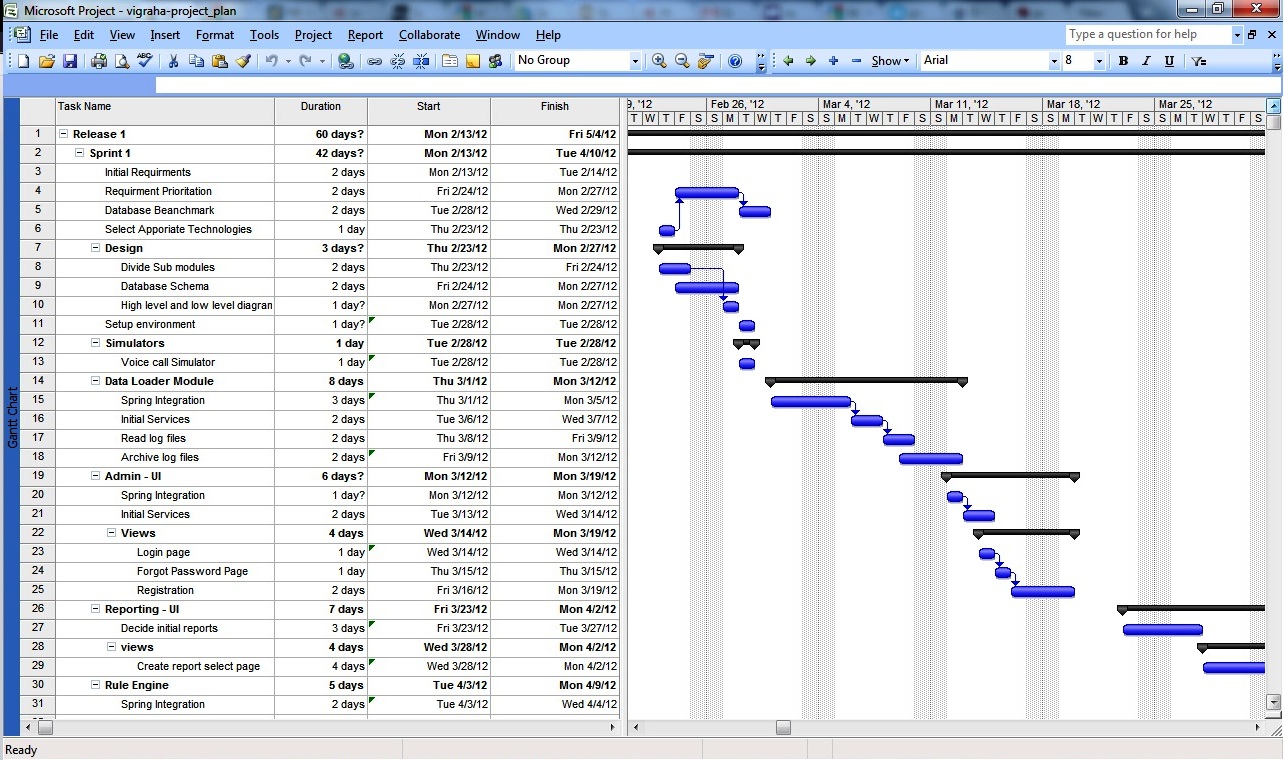
**Database**

Write data to database

Fire process

**Figure 2**

The project plan for the year is as follows:



**Figure 3**

Work Breakdown Structure:

**Vigraha High Volume CDR Analyzer**

Analyzer

Administrator UI

Maintain log files

Report UI

Create stimulators

Send records to database

Register and log administrator

Get requirements

Generate rules

Execute rule

Analyze records

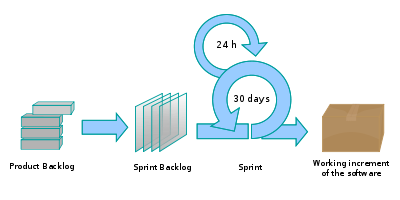
Send SMS

Get requirements

Generate report

Figure 4

Because of the projects delicate and unfamiliar nature, we have decided to employ the scrum approach for developing the system. “Scrum is a form of agile project management” [1]. This model is based on sprints, or short time periods, of 1 to 4 weeks. Within this period of time the development team will work on a portion of the project. And at the end of each sprint the team will meet the client and discuss about any changes that need to be done. If everything is according to the client’s needs we will proceed with the next sprint.

[](http://en.wikipedia.org/wiki/File:Scrum_process.svg)

**Figure 5**

Initial User Stories

In scrum entire system need to describe with user stories. Following are the user stories what we identified.

1. Read log files
2. Archive Log files
3. Summery table uploading schedules
4. Spring Integration
5. Initial Services
6. CAS integration with Single Sign On
7. Open ID Integration
8. Create rules based on programs
9. Search rule
10. View and edit created rules
11. Data analyze with map reduce
12. Configure properties
13. Initial Reports
14. Identify Drill down reports and charts
15. Identify necessary indexes
16. Spring Birt integration
17. Test cases
18. Concurrent access
19. Security
20. SSL for User interfaces

Here are the initial user stories. After Identify user stories we need to estimate those above stories.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| User Stories | | | | | | | | | | | | | | | | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| Rajith | 3 | 3 | 5 | 8 | 13 | 8 | 5 | 3 | 3 | 3 | 13 | 5 | 8 | 5 | 3 | 13 | 5 | 0 | 0 | 0 |
| Lasantha | 3 | 3 | 3 | 8 | 13 | 5 | 8 | 5 | 3 | 3 | 13 | 5 | 5 | 5 | 5 | 13 | 5 | 0 | 0 | 0 |
| Thejani | 3 | 3 | 3 | 13 | 13 | 8 | 8 | 8 | 3 | 3 | 13 | 5 | 8 | 5 | 3 | 13 | 5 | 0 | 0 | 0 |
| Naveen | 3 | 3 | 5 | 8 | 13 | 8 | 8 | 5 | 3 | 3 | 13 | 5 | 8 | 5 | 3 | 13 | 5 | 0 | 0 | 0 |
| Final Extimation | 3 | 3 | 5 | 8 | 13 | 8 | 8 | 5 | 3 | 3 | 13 | 5 | 8 | 5 | 3 | 13 | 5 | 0 | 0 | 0 |

Software Development Requirements

* Net beans – development IDE, we will be designing with Java.
* MySQL – the main database.

Graphics

* Microsoft Project 2007 – Used to make the Gann chart.
* Photoshop – designing purposes of the system.

Technologies

* Apache Hadoop – Used by the Rule Engine for analyzing records.

We have to give a special emphasis on Apache Hadoop. “***The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using a simple programming model. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-avaiability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-availabile service on top of a cluster of computers, each of which may be prone to failures.”***[2]

Anticipated Conclusion

The project will be most useful for mobile companies and external companies willing to advertise through use of messages. Out main priority for the project is handling the number of files that are created every day. We expect this task to be done with no problem and perform its tasks without exception.

Description of Personal and Facilities

Each member of the group is assigned with their own component of the system, evenly weighted. So everyone has the same amount of responsibilities and work. The components are independent so every member can work individually on their part and integrate together afterwards.

|  |  |
| --- | --- |
| Name | Tasks |
| Naveen | * Create stimulators for CDR files. * Develop the database to store daily customer details. * Write data from files to database. * Move read files to a different place. |
| ThejaniDineshika | * Develop Graphical User Interface for administrator. * To register to the system. * To log to the system * To change pass word * Develop Graphical User Interface for the options. * Promotions. * Loyalty programs. * Tenure program. * Generate the rules. |
| RajithDelantha | * Analyze the customer records by hadoop integration. * Execute the rules when necessary time. * At same time when the rule generates may not need to execute it. It may be done in next day. |
| LasanthaPerera | * Develop to generate reports * If the brand companies or mobile company need the current reports the system able to provide them. * The system able to provide history reports when necessary. Here uses the drill down reports.   To generate reports has to filter them according to the requirements. Has to check whether the SMS receive to the mobile users. |

References

[1] “Scrum (development)”, <http://en.wikipedia.org/wiki/Scrum_(development)>

[2] “Apache Hadoop”, <http://hadoop.apache.org/>

Appendix

Appendix A: list of Acronyms and Abbreviations

CDR – Caller Detail Record

SMS – Short Message Service

LBS – Location Base System

GPRS – General Packet Radio Services

USSD – Unstructured Supplementary Service Data

Appendix B: Diagrams and Figures

Figure 1

The High Level Diagram for the entire system. It shows how each component is connected to each other.

Figure 2

Shows the Data Loader component of the system.

Figure 3

The projected plan for the system.

Figure 4

Work break structure shows the main components and the functions that fall under it.

Figure 5

The scrum project management method. Shows how the scrum process works.