ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"Uso de la plataforma Pig sobre Hadoop como alternativa a una RDBMS para el análisis de datos masivos. Prueba de concepto utilizando registros de Detalles de Llamadas"

INFORME DE MATERIA DE GRADUACIÓN

Previo a obtención del Título de:

- **Ingeniero en Computación especialización Sistemas Tecnológicos**
- **Ingeniero en Computación especialización Sistemas de Información**

Presentado por:

Romeo Elías Cabrera Arévalo

Fabricio Felipe Medina Palacios

GUAYAQUIL – ECUADOR

2010

AGRADECIMIENTO

Agradecemos a Dios, a nuestras familias, profesores y amigos que nos proporcionaron fuerzas, ayuda, conocimiento y apoyo para seguir adelante. A la MSc. Cristina Abad por su apoyo y colaboración para la culminación de este proyecto.

DEDICATORIA

A Dios

A nuestros padres

A nuestros familiares

A nuestros amigos

TRIBUNAL DE SUSTENTACIÓN

MSc. Cristina Abad
PROFESOR DE LA MATERIA

MSc. Rebeca Estrada
PROFESOR DELEGADO DEL DECANO

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, nos corresponden exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

(Reglamento de Graduación de la ESPOL)

Fabricio Medina

Romeo Cabrera

RESUMEN

El propósito de este proyecto de materia de graduación es el de crear un sistema capaz de procesar cantidades masivas de datos de detalles de llamadas y mensajes generadas por una empresa de telefonía celular, usando un sistema de computación distribuida.

En el capítulo 1, se identifican los objetivos del proyecto; así como también se especifica el alcance.

En el capítulo 2, se realiza el análisis del problema presentando el marco teórico de la tecnología a emplearse, y también explicando la arquitectura del sistema existente actualmente en la empresa objeto de este estudio, al cual se acoplará y en parte reemplazará el sistema desarrollado.

En el capítulo 3, se trata sobre el diseño de la solución. Primeramente se presentan las herramientas escogidas para implementar el sistema y la arquitectura del mismo. Luego se detalla el formato de archivos a ser procesados y su esquema de almacenamiento. Se finaliza con un ejemplo de código comentado.

En el capítulo 4, se describe la interfaz de usuario del sistema, y las opciones con las que cuenta.

En el capítulo 5, se presentan los resultados de rendimiento de las pruebas realizadas. Además se presenta una comparación de los costos de la tecnología usada actualmente versus el nuevo sistema propuesto.

Finalmente, se presentan las conclusiones obtenidas en el desarrollo del sistema; también recomendaciones en cuanto a mejoras y herramientas complementarias al alcance realizado.

ÍNDICE GENERAL

Pág.

RESUMEN V

ÍNDICE GENERAL VII

ÍNDICE DE FIGURAS IX

ÍNDICE DE TABLAS X

INTRODUCCIÓN XI

1. PLANTEAMIENTO DEL PROBLEMA 1

1.1 Objetivos 1

1.2 Alcance 1

2 ANÁLISIS DE LA SOLUCIÓN 3

2.1 Marco teórico 3

2.2 Arquitectura del sistema existente de recolección de CDRs 5

2.3 Procesos de CDRs de voz (registros de llamadas) 6

2.4 CDRS de SMS (registros de mensajes de texto) 10

3 DISEÑO DE LA SOLUCIÓN 11

3.1 Marco teórico 11

3.2 Archivos CDRs 13

3.3 Arquitectura general de la solución 15

3.4 Esquema de directorio para los archivos CDRs 16

3.5 Ejemplo de código PIG utilizando solución: 18

4 DESCRIPCIÓN DE PANTALLAS E INTERFAZ CON EL USUARIO 20
4.1 Consultas predeterminadas 21
4.2 Ingreso de consultas (por teclado, por archivo) 23
4.2.1 Ingreso de consulta por teclado 23
4.2.2 Ingreso de consulta por archivo de texto 24
4.3 Consulta de requerimientos ingresados y revisión de resultados 26
5 RESULTADOS Y ANÁLISIS COMPARATIVO 33
5.1 Resultados de pruebas. 33
5.2 Costos de tecnología actual. 34
5.2.1. Costos de licencia y soporte Oracle 35
5.2.2. Costos de licencia y soporte HP 36
5.3 Costos del nuevo esquema. 36
5.3.1 Almacenamiento: 36
5.3.2 Transferencia de información: 37
5.3.3 Procesamiento: 37
CONCLUSIONES Y RECOMENDACIONES 39
Conclusiones. 39
Recomendaciones: 40
REFERENCIAS BIBLIOGRÁFICAS 42

ÍNDICE DE FIGURAS

Pág.

Figura 2.1: Arquitectura actual 5

Figura 3.1: Arquitectura general de la solución 15

Figura 3.2: Esquema de directorios 17

Figura 4.1: Ventana de consultas predeterminadas 21

Figura 4.2: Ventana de selección de rangos de fechas 22

Figura 4.3: Ventana de confirmación de consulta predeterminada 22

Figura 4.4: Ventana de selección de consultas por teclado o archivo 23

Figura 4.5: Ventana de ingreso de consultas por teclado 24

Figura 4.6: Ventana de ingreso de consultas por archivo. 25

Figura 4.7: Ventana de consulta de requerimientos ingresados al sistema .. 26

Figura 4.8: Ventana de presentación de resultados 28

Figura 4.9: Ventana de detalle de requerimientos 29

Figura 4.10: Ventana de presentación de resultados en Gráfico Lineal 30

Figura 4.11: Ventana de presentación de resultados en Gráfico de barras .. 31

Figura 4.12: Ventana de presentación de resultados en Gráfico de Pie 32

ÍNDICE DE TABLAS

Pág.

TABLA I: Descripción de campos de archivos de registros de llamadas 13

TABLA II: Resultado de pruebas con variación de nodos 34

TABLA III. Costos de licencia Oracle Enterprise Edition 35

TABLA IV: Costos de uso de nodos y tecnología AMAZON 38

INTRODUCCIÓN

La telefonía celular es una industria que desde sus inicios se ha expandido de gran manera, llegando a tener hoy en día una alta penetración. Y esto no solamente comprende la comunicación por voz, sino también una amplia gama de servicios de valor agregado como son los mensajes de texto (SMS), mensajes multimedia, y navegación en Internet.

La cantidad de información generada al registrar y tasar todos los eventos de uso del servicio, realizados por todos los usuarios de una empresa de telefonía determinada, implica un espacio de almacenamiento del orden de los cientos de gigabytes en un día, y de decenas de terabytes en un mes. [1].

El enfoque común utilizado por estas empresas es utilizar software desarrollado in-house o por terceros que tome esta información y la almacene de manera estructurada en bases de datos para procesos de reportería y facturación [1].

La desventaja de este enfoque es que en estas bases de datos, el esquema utilizado y los índices creados para las tablas, están fuertemente relacionados con la aplicación que accede a la información. Esto dificulta el poder realizar consultas ad-hoc. Otra desventaja es que es muy costoso mantener información del orden de los cientos de terabytes en una base de datos, lo cual obliga a que únicamente se almacenen pocos meses de información en ella. Esto dificulta la realización de procesos de minería de datos o análisis de tendencias para un largo período de tiempo [2].

El propósito de este proyecto de materia de graduación es el de crear un sistema capaz de procesar cantidades masivas de datos de detalles de llamadas y mensajes generadas por una empresa de telefonía celular, usando un sistema de computación distribuida.

CAPÍTULO 1

1. PLANTEAMIENTO DEL PROBLEMA

1.1 Objetivos

1. Comprobar la escalabilidad y adecuación al uso de Pig-Hadoop para el procesamiento de cantidades masivas de registros.

2. Comparar la razón costo/rendimiento entre el uso de Pig-Hadoop en un clúster en la nube contra el uso de un RDBMS comercial para procesar registros masivamente.

3. Demostrar la facilidad de crear consultas ad-hoc usando Pig para diversas y cambiantes necesidades de análisis de información.

1.2 Alcance

1. Instalar una plataforma basada en Pig sobre Hadoop para el procesamiento de registros de detalles de llamadas.

2. Configurar esta plataforma para que sea ejecutada sobre Amazon EC2.

3. Desarrollar scripts en Pig que realicen análisis de esta información a nivel de: Reportería, Estadísticas, Tendencias, etc.

4. Comparar esta solución frente al uso de una herramienta comercial (Oracle) en una empresa de telefonía celular local teniendo en cuenta valores de licencia, servidores, mantenimiento, tiempos de procesamiento, etc.

5. Desarrollar un sencilla interface Web para poder ingresar y/o subir scripts en Pig para esta plataforma, y para realizar una visualización de los resultados.

CAPÍTULO 2

2 ANÁLISIS DE LA SOLUCIÓN

2.1 Marco teórico

Se determina la necesidad de encontrar una solución informática que sea:

1. Escalable
2. Económica
3. Eficiente
4. Confiable

Como solución al problema planteado, se determina el utilizar Apache Pig, el cual es una plataforma que utiliza Hadoop que a su vez es una implementación del paradigma MapReduce de procesamiento masivo de datos.

MapReduce es un paradigma de software, creado originalmente en Google para soportar la computación distribuida en datos a gran escala realizada sobre clústeres de computadores. En este esquema, dos pasos son llevados a cabo para obtener un resultado. Primeramente el "Map", en el cual un nodo primario del clúster toma los datos de entrada, los subdivide en subtareas más pequeñas, y los distribuye a nodos trabajadores. Luego en el paso "Reduce", el nodo primario toma todas las respuestas de cada subtarea y las combina de tal manera que se obtiene el resultado esperado al problema original [3].

Hadoop es un proyecto de software libre de la Apache Software Foundation que implementa el paradigma MapReduce. Hadoop consiste en el "Hadoop Core", el cual provee soporte para sistemas de archivos distribuidos (o se puede utilizar su sistema de archivos nativo: el HDFS) y de la característica "Rack Awareness" (para tomar decisiones de tráfico de datos, dependiendo de localización física). Sobre esta capa, funciona el motor de MapReduce (una implementación propietaria del paradigma originado en Google), el cual consiste de un Job Tracker y varios Task Trackers, los cuales coordinan las tareas que son ejecutadas por el clúster [4].

Apache Pig es una plataforma de software que provee un lenguaje de alto nivel declarativo y secuencial para realizar análisis de datos. Pig transforma sentencias escritas por el usuario en este lenguaje propio (Pig Latin), en código MapReduce que es manejado por la plataforma Hadoop [5].

2.2 Arquitectura del sistema existente de recolección de CDRs

Los CDR son registros informáticos generados por una central telefónica, los cuales contienen detalles de las llamadas o mensajes que han pasado a través de ella.

La Figura 1 muestra el esquema general del sistema de recuperación y almacenamiento de CDRs de una empresa de telefonía del país. Los CDRs son generados cada 5 minutos por el proveedor de la central telefónica o de mensajes, y son depositados en 4 servidores principales locales conocidos como nodos o centrales.

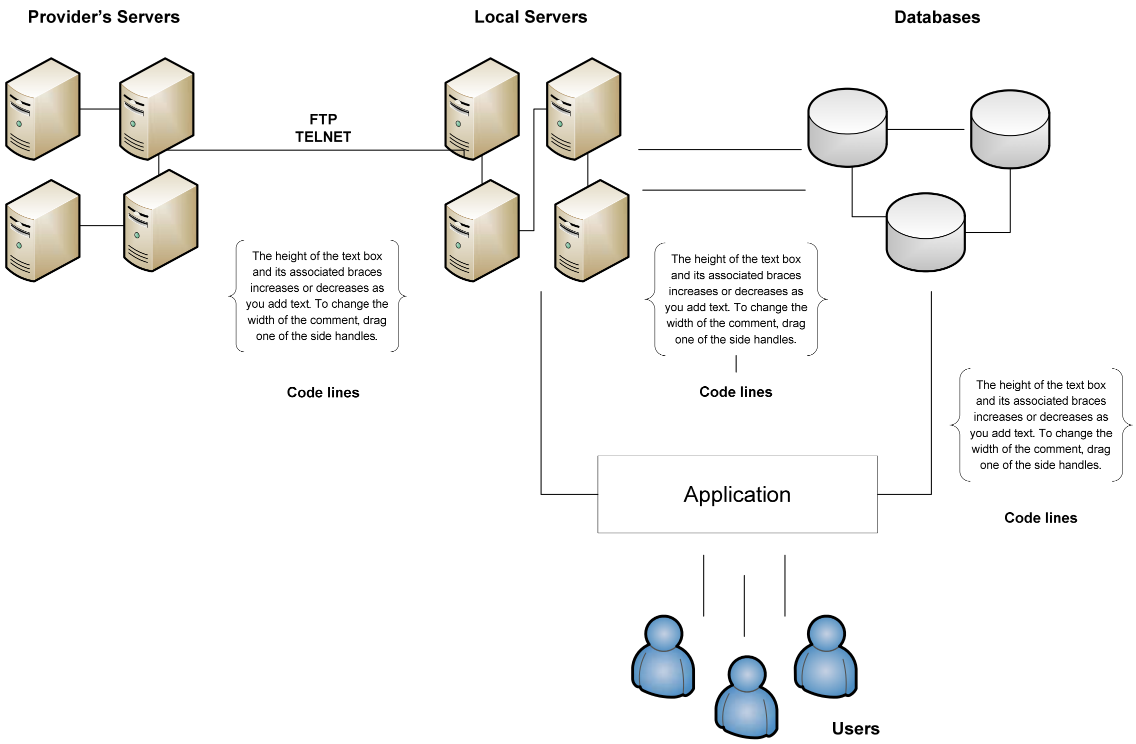

Figura 2.1: Arquitectura actual

La generación de CDRs, ejecución transacciones y eventos está dividida en cada una de estas centrales de la siguiente manera:

- Central 1: Líneas de suscriptores cuyos 2 últimos dígitos finalizan en el rango de 00 – 24.
- Central 2: Líneas de suscriptores cuyos 2 últimos dígitos finalizan en el rango de 25 – 49.
- Central 3: Líneas de suscriptores cuyos 2 últimos dígitos finalizan en el rango de 50 – 74.
- Central 4: Líneas de suscriptores cuyos 2 últimos dígitos finalizan en el rango de 75 – 99.

Para el proyecto actual se decide analizar dos básicos tipos de CDRs: CDRs de voz (registros de llamadas) y CDRs de SMS (registros de mensajes de texto).

2.3 Procesos de CDRs de voz (registros de llamadas)

Los CDRs de voz son aquellos que registran todos los eventos de llamadas salientes de un suscriptor. Registran detalles importantes como:

- Identificador o número de la línea que realiza la llamada
- Identificador o número de la línea que recibe la llamada
- Fecha y hora inicial de la llamada
- Ubicación de celda de llamada saliente
- Ubicación de celda de llamada entrante
- Duración de la llamada
- Costo de la llamada

Procesos automáticos toman diariamente los CDRs depositados en las centrales por el proveedor y los transfieren al un servidor repositorio (Colector) donde son procesados y cargados de forma estructurada en tablas de una base de datos Oracle.

Esta transferencia y procesamiento se realiza de manera en-línea y por lotes (batch).

El procesamiento en-línea se realiza por medio de procesos ejecutados por medio de un crontab disparados cada 5 minutos, mismos que se encargan de conectarse vía FTP a las centrales y tomar los CDRs generados no cargados previamente, y transferirlos al Colector principal, luego los mismos son subidos a la tabla de CDRs del día, manteniéndose un detalle de llamadas actualizado durante la jornada.

El procesamiento por lotes es realizado durante horas no laborables, mismo que se realiza una sola vez con el propósito de reprocesar las tablas para descartar CDRs perdidos o no registrados en las mismas.

Los pasos que se siguen para dicho reproceso son los siguientes:

1. Transferencia por lotes de CDRs: Realizado vía FTP, inicia a las 00:20 del día y dura alrededor 50 minutos en promedio.

2. Verificación servidores vs. plataforma externa: Se realiza control de calidad de cantidad de CDRs transferidos a Servidor destino y cantidad de CDRs generados en las centrales.

3. Generación de archivo consolidado. Se realiza un archivo conjunto de todos los CDRs del día anterior, este archivo final es el que se cargará en su totalidad a las tablas de Oracle. Este proceso toma aproximadamente 30 minutos.

4. Truncado de tabla cargada durante el día. Aproximadamente 1 minuto en promedio.

5. Finalmente se realiza la carga por medio de un SQL Loader a la tabla de llamadas, con esto se completa el reproceso. Esta carga dura aproximadamente 50 minutos.

Las tablas se manejan con un índice mismo que fue creado para poder realizar con una mayor facilidad y rapidez búsquedas. Este índice es de suma importancia al realizar detalles de llamadas a través de aplicativos.

Los campos indexados son: SUB_ID, DATE y TRANSACTION_TYPE en ese orden (teléfono que inicia el evento, fecha del evento y tipo del evento).

Cada tabla ocupa un espacio de aproximadamente 6 GB en disco y los índices correspondientes alrededor de 1.5 GB. La Empresa ha determinado una política de almacenamiento de tablas de llamadas de 90 días en sus servidores de Producción en este caso en el Colector principal. Y el departamento de DBA diseñó el esquema de almacenamiento en un tablespace para datos e índices de dichos 90 días.

En la actualidad el tamaño de tablespace para data e índices de 90 días es de 670 GB y el rango de disponibilidad en el mismo es del 6% a 7%.

2.4 CDRS de SMS (registros de mensajes de texto)

Los CDRs de SMS registran cada evento, ya sea de envío o recarga de mensajes de texto de un suscriptor en la plataforma. Los CDRs son generados de igual forma en las centrales siguiendo el esquema mencionado previamente. El proceso que realiza esta tarea es similar a la carga en línea de CDRs de Voz.

El DBA realiza el dimensionamiento diario de cada tablespace en un datafile diferente. El datafile de datos pesa aproximadamente 4.9GB.

Estas tablas constan de dos índices importantes para búsquedas a nivel de aplicativo y base de datos Los campos indexados son:

- NUMBER_CALLING (teléfono que envía el mensaje) en índice 1
- NUMBER_CALLED, RED_DESTINO (teléfono receptor del mensaje y red destino) en índice 2

Los datafiles para estos índices ocupan un espacio de almacenamiento entre 1.7 y 1.2 GB por día.

La política de almacenamiento de la empresa para los registros de SMS es también de 90 días. Se tiene entonces una totalidad de 702 GB de espacio de almacenamiento necesarios en los sistemas de producción para esta información.

CAPÍTULO 3

3 DISEÑO DE LA SOLUCIÓN

3.1 Marco teórico

Para realizar una prueba de concepto, se configuró una instalación de Hadoop levantada sobre un clúster de computadores levantados bajo demanda con un servicio de IaaS (Infrastructure as a Service).

Se escoge utilizar como archivos de entrada archivos comprimidos en formato BZ2. Pig soporta el manejo de archivos comprimidos en este formato. La ventaja de utilizar BZ2, es que estos archivos son "divisibles", es decir, que es posible el acceder a cualquier punto del archivo y comenzar a leer desde esa ubicación en adelante. Esto facilita el dividir la tarea de procesar un sólo archivo comprimido entre varios Mappers [6].

Estos archivos de entrada son CDRs (Call Detail Records) de eventos de mensajes (SMS) generados como archivos de texto. Cada línea del archivo equivale a un registro, con sus campos separados por comas.

Los servicios web de Amazon (Amazon Web Services) son una colección de servicios de computación remota (en las nubes) ofrecidos sobre Internet por Amazon.com. Estos servicios facilitan el desarrollo de aplicaciones distribuidas. Los servicios utilizados en este proyecto son: Amazon S3 y Amazon EC2.

Amazon Simple Storage Service (S3) es un servicio de almacenamiento en Internet. Provee una interfaz simple de servicios Web que puede ser utilizada para almacenar y recuperar cualquier cantidad de datos en cualquier momento, en cualquier lugar de la Web. Está diseñado para facilitar el diseño de aplicaciones escalables en Internet.

Amazon Elastic Cloud Computing (EC2) es un servicio web que provee una flexible capacidad de procesamiento para aplicaciones en la nube. Este servicio provee un ambiente de cómputo virtual, sobre el cual pueden usarse servicios Web para levantar instancias de una variedad de sistemas operativos cargados con un ambiente de aplicaciones predefinidas.

3.2 Archivos CDRs

A continuación, se presenta un ejemplo de una línea de este archivo, correspondiente al registro de un mensaje SMS enviado.

```
10888,59390730123,6-29-2009,86067,6-30-
2009,17667,202,304,1222F832640858180069073012300980,
1436,0,0,0,319,,108,0,0,0,701,1002,101,-1,0,
,0,9,35,9/28/2008 17:55:0,1/1/1900 0:0:0,9/28/2008
17:55:0,0,0,0,0,,0,0,0,0,0,0,1,2,1,350,16,50,-
1,0,59385818006,740010115532771,59397995028,2,1,
```

A continuación se presenta el formato del archivo, el cual indica qué representa cada campo del registro [7]:

TABLA I: Descripción de campos de archivos de registros de llamadas

1	Sequence Number	28	Suspended Date
2	Subscriber ID Cdr Rate	29	Frozen Date
3	Cdr Seconds Source Date	30	Group type
4	Source time	31	Group Id
5	Service type	32	Rule Discount
6	Transaction Type	33	Amount
7	Correlation Id	34	Rule Discount units
8	Transaction Major Num	35	Rule Disount Name
9	Transaction Minor num	36	Discount amount F&F
10	Source Type	37	Discount Scheme Id 0
11	Source Info1	38	Discount Amount 1
12	Source Info2	39	Discount Scheme Id 1
13	Rate Name	40	Discount amount 2

14	Tariff Plan Id	41	Discount Scheme Id 2
15	System Tax %	42	Account type
16	Location Tax %	43	Account status
17	Service Tax %	44	Balance type
18	Billing Event Id	45	Balance def Id
19	Time Band Group Id	46	Balance amount
20	Service Category Id	47	Balance units
21	Operator Id	48	Amount debit / credit
22	Operator Transact type	49	Reservation count
23	Account Number	50	Bnumber
24	Subscriber Class	51	IMSI - A Number
25	Profile Id	52	Sub Location
26	Options Array	53	Network Type
27	Service Fee Date	54	NumMessagesExtraBalanceFields

La operadora celular que registra esta información, procesa los datos de cuatro centrales de procesamiento. La información generada por una sola de estas centrales durante un día equivale a 1.2 GB de información sin comprimir, en total 5 GB. Una vez comprimido en formato BZ2, el total de archivos ocupan un espacio de almacenamiento de 3 GB.

3.3 Arquitectura general de la solución

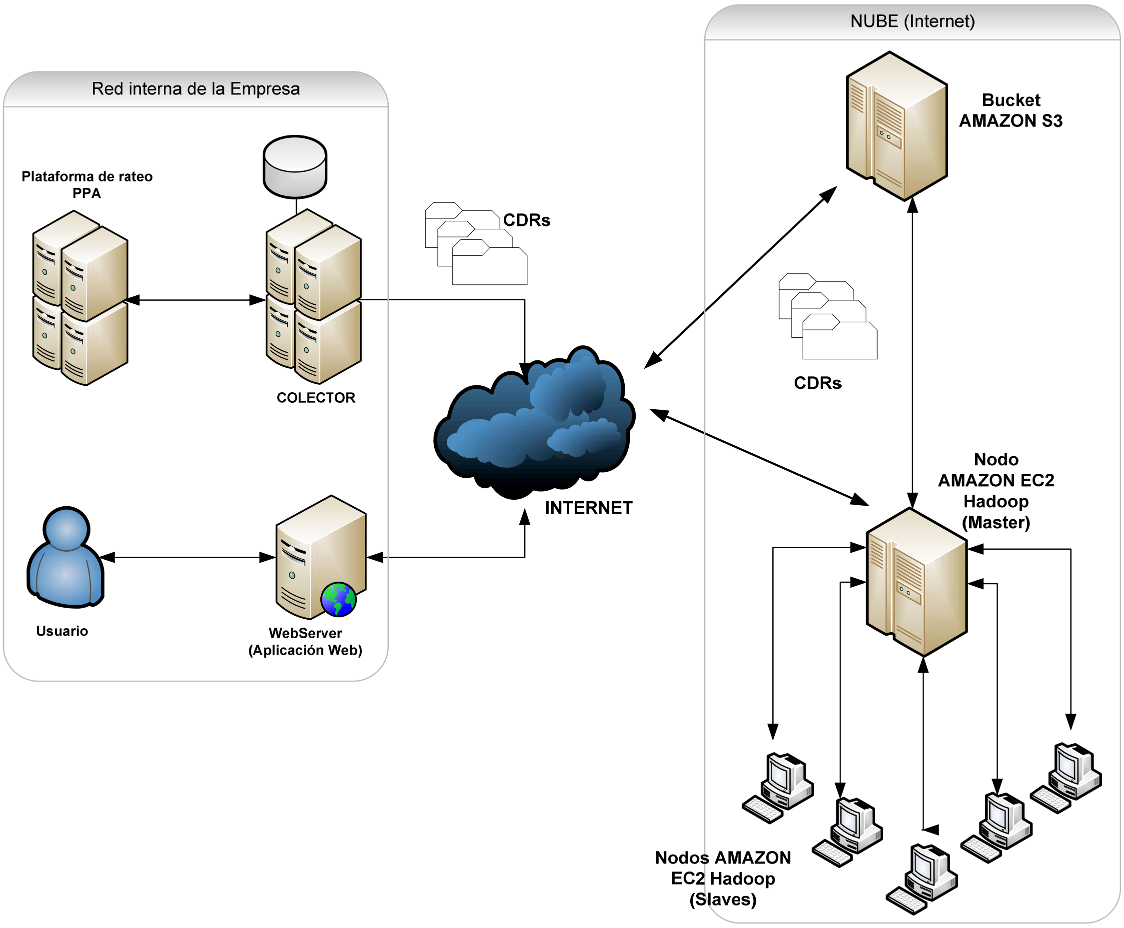

Figura 3.1: Arquitectura general de la solución

Componentes:

1. En tiempo real, se generan archivos CDRs en la plataforma de rateo prepago, y son transferidos al servidor que sirve como colector de información. Se genera un archivo aproximadamente cada 5 minutos, con una longitud promedio de 300 KB.

2. Shell script que se ejecuta diariamente por un crontab, y que consolida todos los archivos generados en un día en un solo archivo, los comprime en formato BZ2, y los transfiere al Bucket (directorio) en Amazon S3.

3. La interfaz con el usuario es una página Web, por medio de la cual se ingresan scripts en el lenguaje Pig. El servidor Web envía este script al nodo maestro de un clúster Hadoop corriendo sobre Amazon EC2.

4. Este clúster de Hadoop obtiene la información del Bucket (directorio) en Amazon S3, mencionado en el punto 2, usando la herramienta s3cmd.

5. La salida del proceso se transfiere a un directorio en el Bucket Amazon S3 para que pueda ser leído desde el servidor Web.

3.4 Esquema de directorio para los archivos CDRs

Con el fin de optimizar el acceso a los archivos de CDRs, se utiliza un esquema de directorios basado en fechas (año, mes y días), y así facilitar las consultas realizadas sobre un rango de tiempo. De esta manera, por ejemplo, los archivos de SMS generados el 14 de julio de 2009, se ubican en la ruta /DATOS/SMS/2009/07/14.

Para acceder a los datos de VOZ de todo el mes de febrero, se especifica como datos de entrada, el directorio /DATOS/VOZ/2009/02

Figura 3.2: Esquema de directorios

3.5 Ejemplo de código PIG utilizando solución:

El siguiente script de PIG, realiza una sumarización de la cantidad total de mensajes enviados por cada uno de los teléfonos y ordenarlo en orden descendente por cantidad de mensajes:

```
A = LOAD 'input' USING PigStorage(',');
Y = GROUP A BY $1;
Z = FOREACH Y GENERATE $0,COUNT(A);
N = ORDER Z BY $1 DESC;
STORE N INTO 'output/sms' USING PigDump();
```

A = LOAD 'input' USING PigStorage(','): Almacena en la colección “A” toda la información que se encuentre en carpetas y subcarpetas del directorio “input”. Pig permite especificar diversos métodos de lectura de información (funciones de carga). En este caso se utiliza la función de carga “PigStorage” la cual convierte cada línea de texto en un registro delimitado por un separador que se indica como parámetro. En este caso el delimitador es la coma (“,”)[8].

Y = GROUP A BY $1: Define una colección “Y” como el agrupamiento de cada registro de acuerdo al campo ubicado en la posición 2 (Pig etiqueta los campos desde 0, por eso se indica $1 en el script). En este caso, la posición 2 corresponde al número de teléfono que envía el mensaje.= FOREACH Y GENERATE $0,COUNT(A); Se define a la colección Z como el resultado de iterar cada elemento de la colección Y generando dos registros. Primeramente, el primer elemento de la colección Y (que es el número de teléfono por el cual se hizo el agrupamiento). En segundo lugar, la cantidad (COUNT) de registros asociados de la colección A.

N = ORDER Z BY $1 DESC: Se define la colección N como el resultado de ordenar en orden descendente la colección Z por su segundo campo, el cual en este caso es la cantidad de mensajes.

STORE N INTO 'output/sms' USING PigDump(): Almacena en disco el resultado de la colección N, dentro de la carpeta “output/sms”. Se define el uso de la función PigDump como método de almacenamiento (almacenamiento en archivo de texto plano).

Al ejecutarse los comandos anteriores, el motor interno de Pig analiza y optimiza el script ingresado y genera uno o más trabajos MapReduce con el fin de obtener el resultado definido en el flujo de sentencias Pig [5].

CAPÍTULO 4

4 DESCRIPCIÓN DE PANTALLAS E INTERFAZ CON EL USUARIO

Con el propósito de automatizar la administración de requerimientos de procesamiento de CDRs y de hacerlo amigable para el usuario final, se desarrolló un aplicativo Web como front-end de la solución presentada en capítulos anteriores, llamado “Consultator versión 1.0”.

Este sistema se desarrolló bajo la plataforma LAMP (Linux, Apache Server, MSQL y PHP) y está compuesto por páginas PHP para la navegación un motor de base de datos MYSQL para la administración de los requerimientos.

El sistema de procesamiento masivo de CDRs realiza básicamente las siguientes operaciones:

1. Realizar consultas predeterminadas

2. Ingresar consultas, ya sea que se ingrese algún script de PIG, o el ingreso de scripts por un archivo de texto.

4.1 Consultas predeterminadas

El aplicativo presenta las siguientes consultas predeterminadas

1. Cantidad de llamadas

2. Cantidad de sms

3. Conteo de llamadas por duración

Figura 4.1: Ventana de consultas predeterminadas

Al escoger uno de las opciones de consulta se mostrará una ventana emergente para poder ingresar el rango de fechas a consultarse.

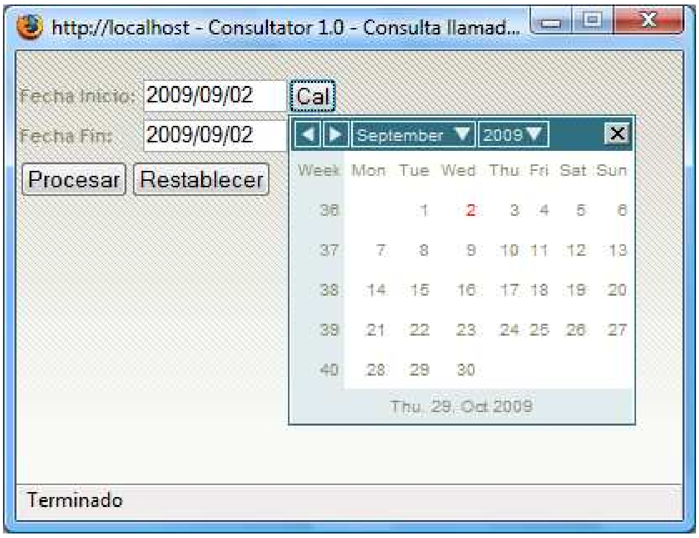

Figura 4.2: Ventana de selección de rangos de fechas

Luego de seleccionar las fechas, debe de dar clic en Procesar y el requerimiento es ingresado. Si el mismo es ingresado correctamente se le mostrará el siguiente mensaje:

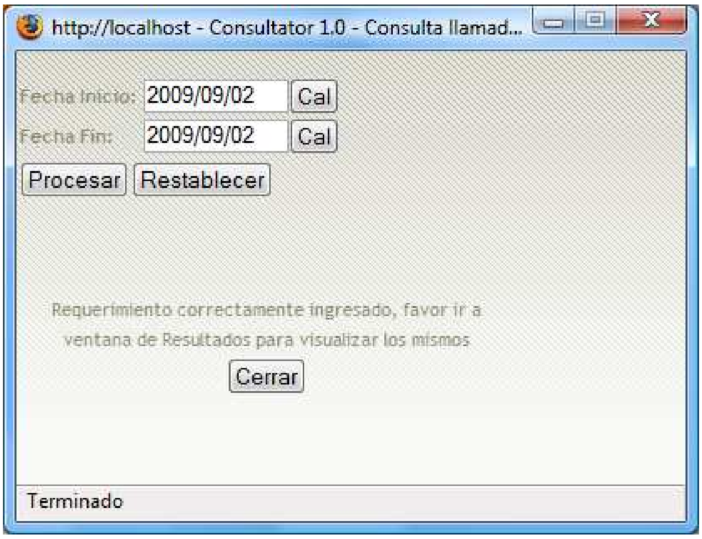

Figura 4.3: Ventana de confirmación de consulta predeterminada

4.2 Ingreso de consultas (por teclado, por archivo)

Luego de accesar al sistema el usuario podrá seleccionar dos tipos diferentes de modos para consultas.

1. Ingreso de Pig scripts por teclado
2. Ingreso de scripts por archivo

Figura 4.4: Ventana de selección de consultas por teclado o archivo

4.2.1 Ingreso de consulta por teclado

Al seleccionar el ingreso de scripts PIG se presentará una pantalla con un cuadro de texto activo para poder ingresar el script manualmente. Luego de ingresar el mismo deberá darse clic en Consultar.

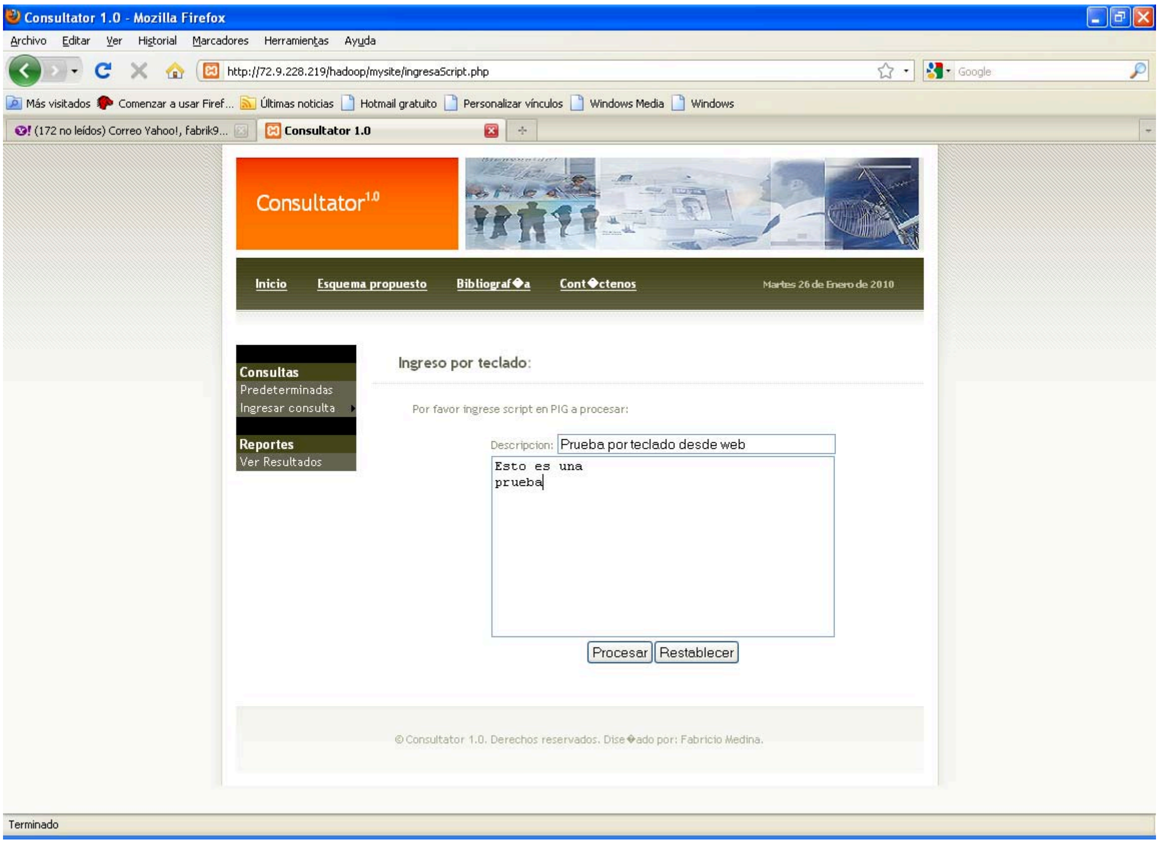

Figura 4.5: Ventana de ingreso de consultas por teclado

4.2.2 Ingreso de consulta por archivo de texto

Al seleccionar el ingreso de scripts por archivo se presenta una pantalla con un botón EXAMINAR mismo que al ser presionado abre un cuadro de diálogo para seleccionar el archivo que contiene la consulta ya estructurada. Luego de ingresar el mismo se hace clic en "Consultar".

Únicamente se permiten archivos de extensión txt y máximo de 1MB de dimensión.

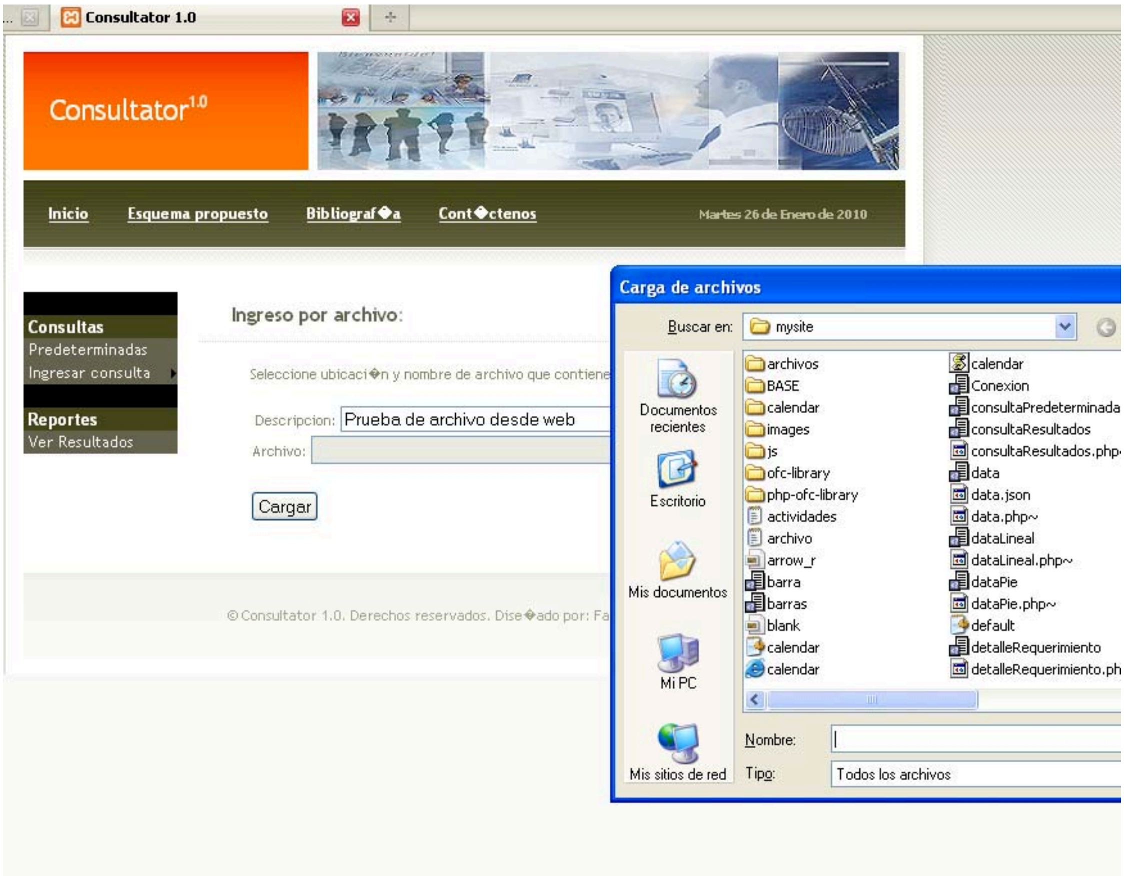

Figura 4.6: Ventana de ingreso de consultas por archivo.

Cada una de estas opciones, luego de ingresar correctamente el requerimiento, da un mensaje de ingreso correcto y añade el código de requerimiento generado automáticamente por el sistema. Este código de requerimiento es el que permite revisar su estado y los resultados esperados.

4.3 Consulta de requerimientos ingresados y revisión de resultados

Luego de ingresar los requerimientos, ya sea predeterminada o ingresada por alguno de los medios disponibles, se puede ver el estado del mismo a través de las ventanas de Reportes. Para esto se debe ir al menú principal y seleccionar la opción "Ver Resultados".

Esta opción presenta la ventana de "Consulta de requerimientos". El usuario debe ingresar el código de requerimiento generado y presentado previamente por el sistema, ingresarlo por teclado en la caja de texto "Buscar requerimiento" y dar clic en el botón Buscar:

Figura 4.7: Ventana de consulta de requerimientos ingresados al sistema

El sistema busca entre los requerimientos previamente ingresados y los presenta en una tabla dinámica. Los campos son:

- Código requerimiento (Código generado automáticamente, único campo para búsquedas)
- Detalle de Requerimiento
- Fecha de requerimiento
- Estado de requerimiento (Procesando o Finalizado)

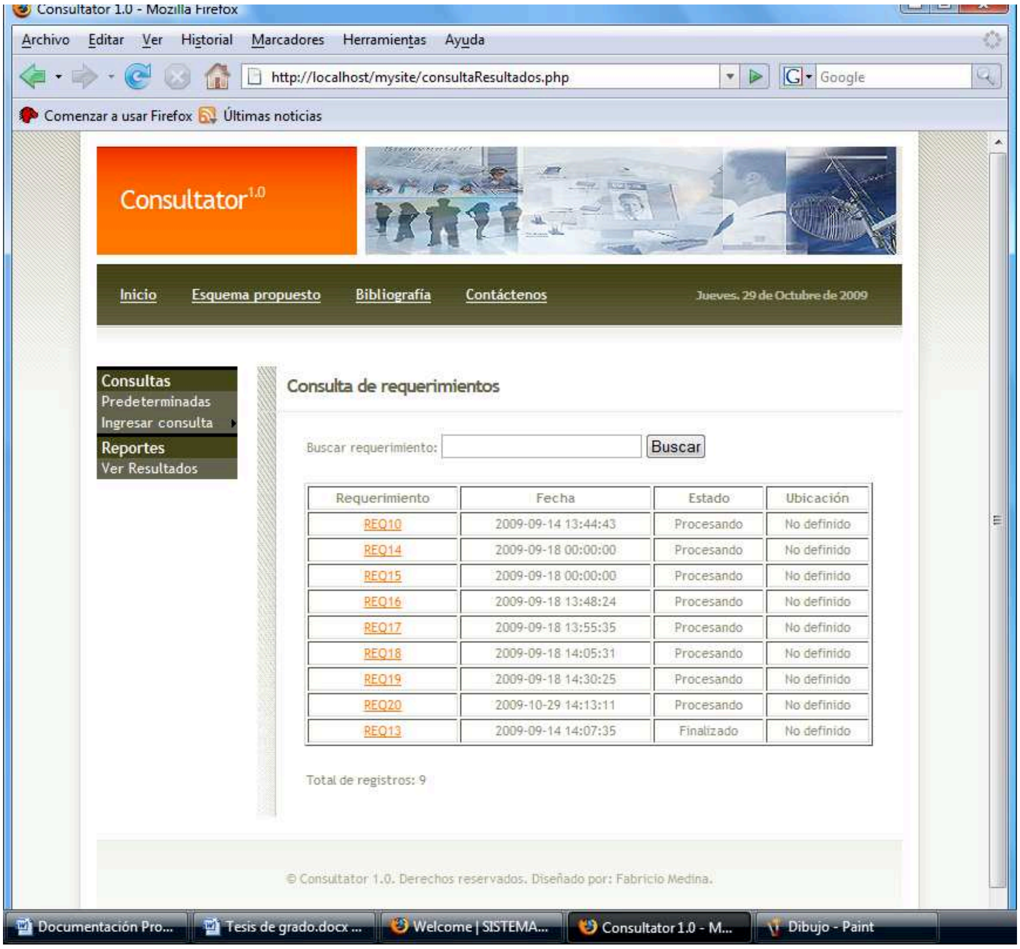

Figura 4.8: Ventana de presentación de resultados

Para revisar los resultados de los requerimientos se dar clic sobre el campo "Requerimiento", mismo que se encuentra en color naranja. Este presenta la ventana de Detalle de requerimientos:

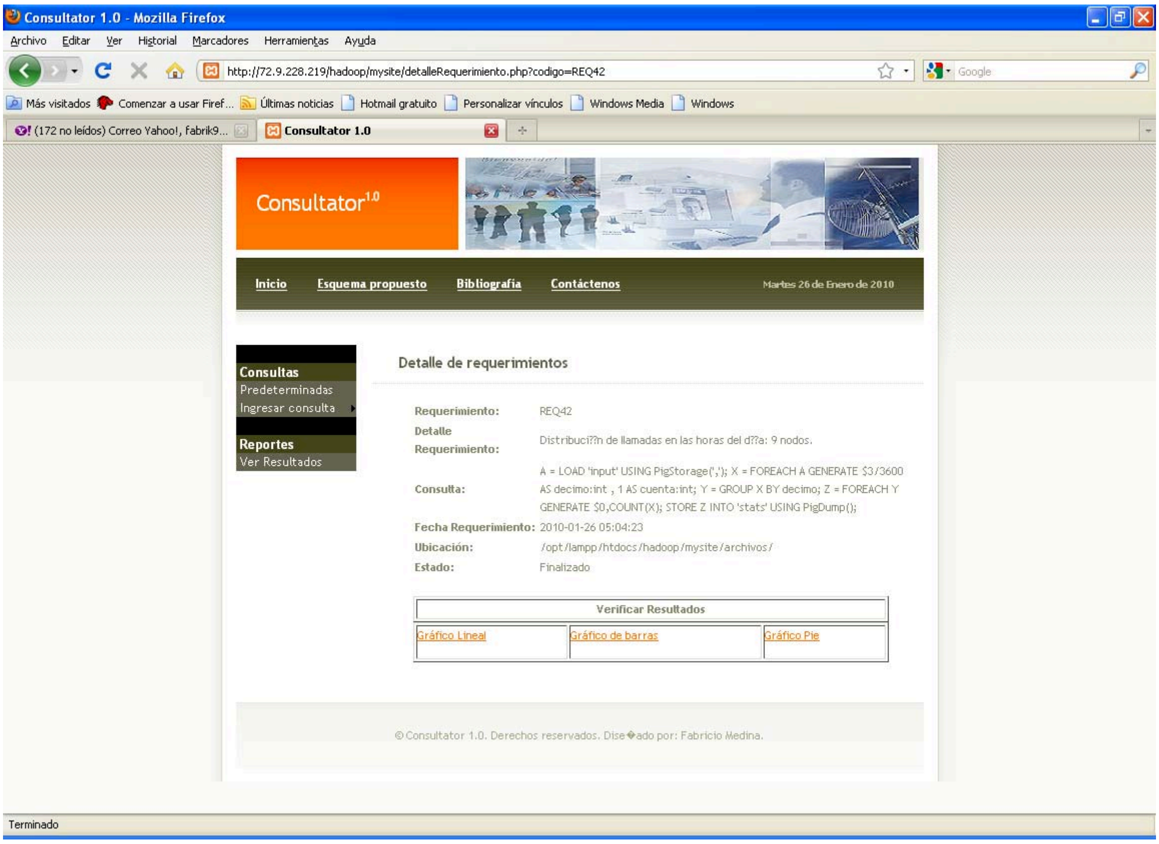

Figura 4.9: Ventana de detalle de requerimientos

La ventana de detalle de requerimientos presenta los datos principales de la solicitud ingresada por el usuario. Estos son:

- Código de requerimiento
- Detalle de Consulta
- Consulta o detalle de script PIG procesado
- Fecha de ingreso del requerimiento
- Ubicación de archivo físico del requerimiento

- Estado de requerimiento

Al final presenta el marco de Resultados, dando las siguientes opciones:

- Resultados presentados como gráfico lineal
- Resultados presentados como gráfico de barras
- Resultados presentados en gráfico de pie o pastel

Los siguientes son ejemplos de los resultados dependiendo del tipo de presentación:

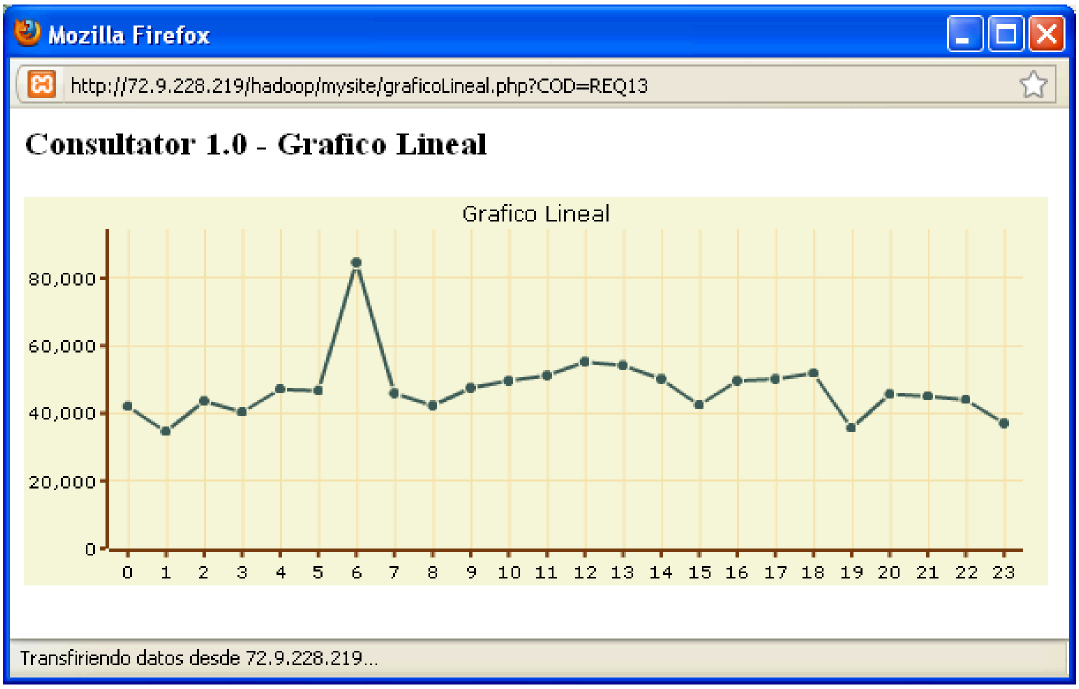

Figura 4.10: Ventana de presentación de resultados en Gráfico Lineal

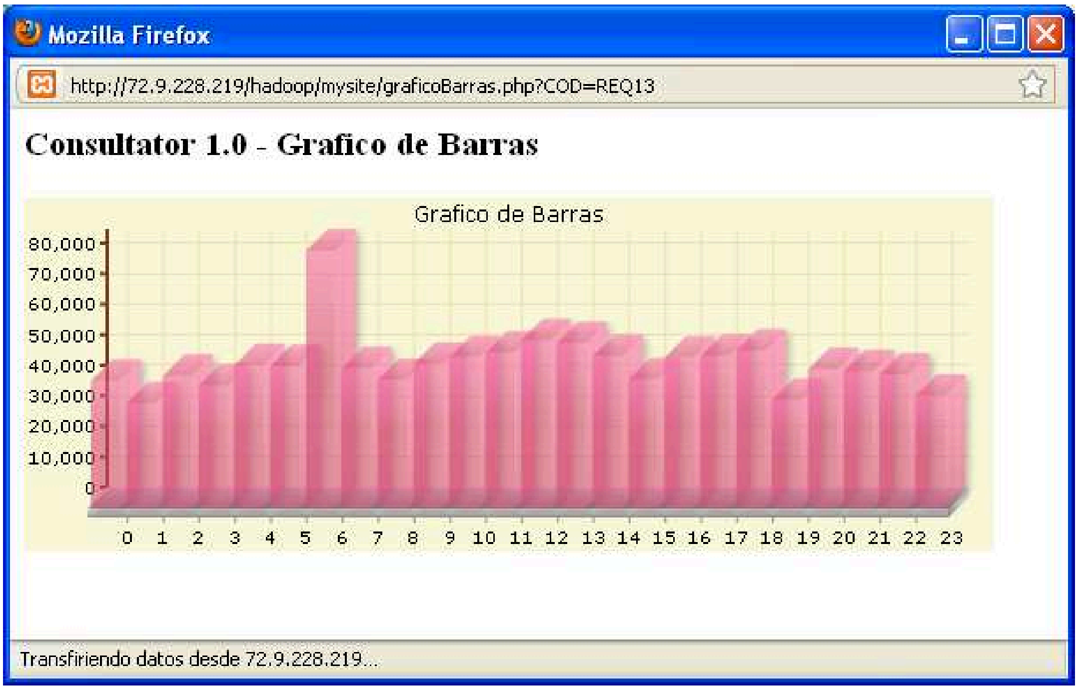

Figura 4.11: Ventana de presentación de resultados en Gráfico de barras

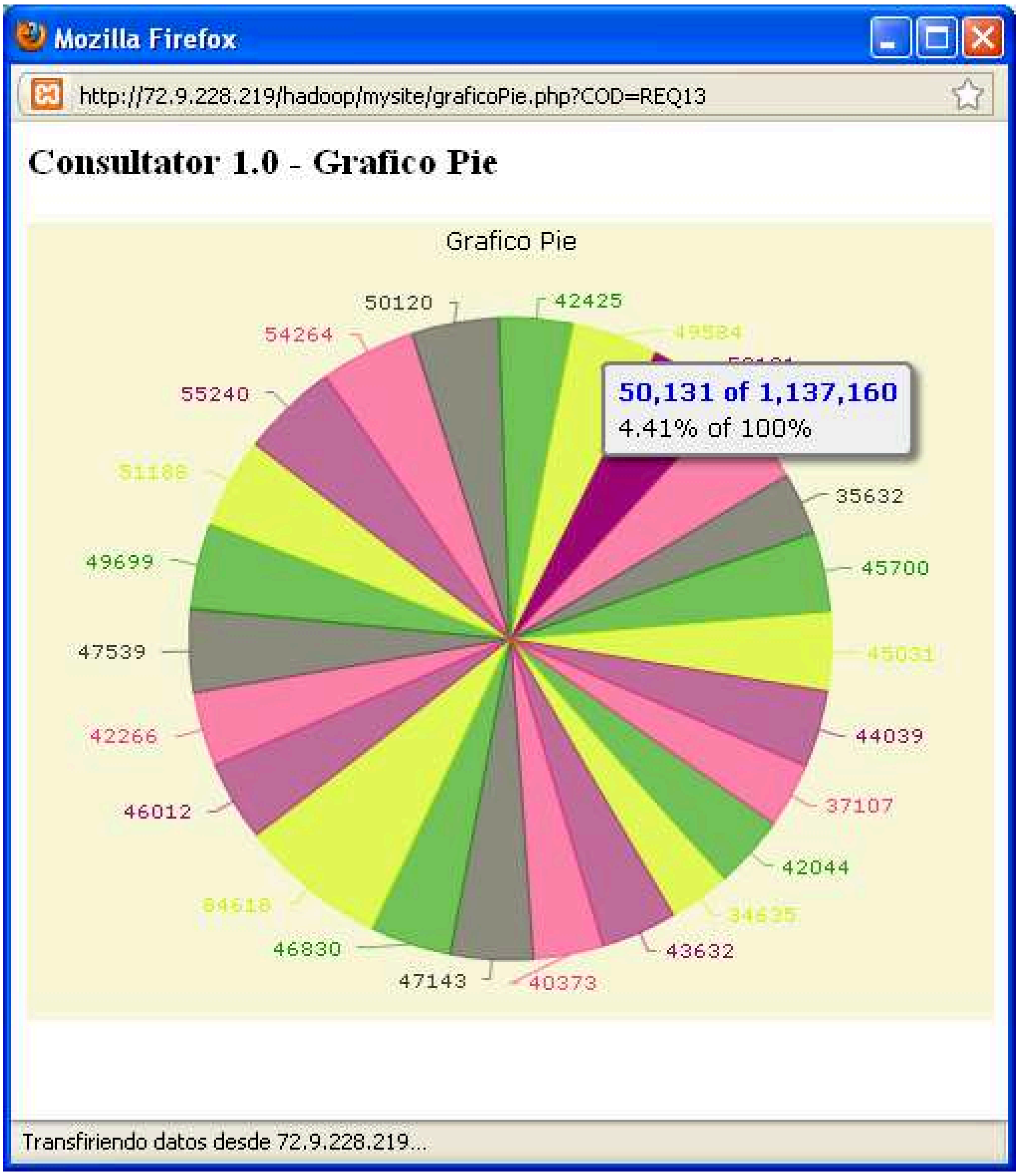

Figura 4.12: Ventana de presentación de resultados en Gráfico de Pie

CAPÍTULO 5

5 RESULTADOS Y ANÁLISIS COMPARATIVO

5.1 Resultados de pruebas.

Estas pruebas se realizaron con archivos CDRs que totalizaban un espacio de almacenamiento 3,15 GB. Estos, al ser comprimidos ocupaban una capacidad de 1,54 GB.

Se realizó una prueba con dos scripts.

1. Sumatoria de SMS. Este script hace un agrupamiento por teléfono de todos los registros de CDRs y genera la cantidad de registros presentes para cada teléfono.

2. Distribución de envío de mensajes. Este script agrupa todos los registros, de acuerdo a la hora del día en el cual el evento fue generado. El resultado del mismo es la cantidad de eventos realizados en una hora del día dada.

Los nodos EC2 con los que se ejecutaron los scripts son del tipo “High CPU On-Demand Instances” de nivel “medium”.

Cada uno de los scripts se lo ejecutó tres veces: con 4, 9 y 16 nodos, siendo los resultados los siguientes:

TABLA II: Resultado de pruebas con variación de nodos

	4 Nodos	9 Nodos	16 Nodos
Sumatoria por teléfono	610	380	368
Distribución horaria	550	315	311

(Tiempo en segundos)

Como se puede observar, al aumentar el número de nodos de 4 a 9, el tiempo de ejecución se reduce a un 37% para el script de sumatoria por teléfono, y en un 42% para el script de distribución horaria. Sin embargo al aumentar el número de nodos de 9 a 16, la mejora es muy poco significativa: 4% para el script de sumatoria y 1% para el script de distribución horaria. Esto se debe a que el número de tareas que se pueden correr en paralelo es limitado y depende de la cantidad y tamaño de archivos, lo cual produce que más allá de cierto número de nodos no se encuentre una mejora significativa en el rendimiento.

5.2 Costos del esquema actual.

La empresa posee una estructura de servidores denominada Super Dome misma que contiene varios servidores denominados principales para la operativa de esta. Entre estos se encuentra el servidor encargado de recolectar los cdrs extraídos de las centrales.

El Recolector principal posee 24 procesadores, se encuentra conectado a la LAN interna y el motor de base de datos que soporta es Oracle.

5.2.1. Costos de licencia y soporte Oracle

La empresa utiliza como base de datos principal Oracle Enterprise Edition versión 10G. Los costos de la licencia, según fuente actualizada son los siguientes:

TABLA III. Costos de licencia Oracle Enterprise Edition

Costo por procesador:	$47.500
No. De procesadores:	24
Costo total de licencia para Servidor Recolector:	**$1’140.000**

Costos de soporte anual:	$10,450

5.2.2. Costos de licencia y soporte HP

El costo del equipo denominado Super Dome, mismo que contiene el servidor Recolector principal bordó el millón de dólares hace aproximadamente dos años.

El costo por mantenimiento y soporte anual del mismo es de $250.000 anuales o $20.000 mensuales. Relacionando la cantidad de servidores internos el costo de mantenimiento del Recolector borda los $3.000 al mes.

5.3 Costos del nuevo esquema.

5.3.1 Almacenamiento:

Amazon cobra una tarifa mensual de almacenamiento en rangos dependiendo de la cantidad de datos almacenados. Considerando una cantidad de 15 GB generados a diario (CDRs de voz, sms y datos) que al ser comprimidos ocupan 7 GB de espacio en disco el costo anual de almacenamiento sería el siguiente de acuerdo a las tarifas actuales [10]:

7 GB/día x $0.15/GB x 365 días = $383,25 anuales a pagar.

Este precio se iría acumulando año a año, es decir, el segundo año se debería pagar $383,25 más los $383,25 del año anterior y así sucesivamente.

5.3.2 Transferencia de información:

Existe un costo del ancho de banda utilizado por la información que es recuperada tanto de los buckets S3 como de los nodos EC2. Sin embargo, el costo de transferencia dentro de la red de Amazon (al recuperar los datos de S3 para procesarlos en EC2 por ejemplo) es de cero [11]. En la actualidad, el costo es de $0.170 por GB lo cual impacta poco en los costos finales, considerando que los resultados que se extraerán como resultado de las ejecuciones de las consultas son reportes sumariados o consolidados. [11]

5.3.3 Procesamiento:

El procesamiento en los nodos EC2 es tarifado con un costo por hora o fracción. Este costo depende del tipo de sistema operativo de los nodos (en este caso, es Linux) y de la características de los mismos (nodos estándar, de alta memoria o de alto poder de CPU).

En las pruebas realizadas se levantaron nodos de tipo “High-CPU Medium instance” los cuales tienen un costo de $0,17 por cada hora o fracción de uso. Estos nodos tienen las siguientes características: 1.7 GB de memoria, 5 EC2 computing units, 350 GB de almacenamiento local y plataforma de 32 bits. El procesamiento de CPU es medido por Amazon en la unidad propietaria “EC2 computing units” la cual es una medida comparativa del poder de procesamiento de un nodo. En la actualidad una EC2 computing unit equivale al poder de CPU de un procesador Xeon 2007 de 1.0-1.2 GHz. [11]

Amazon también provee un esquema de tarifación llamado “Reserved Instances” la cual provee una tarifa por hora rebajada previo al pago de un valor anual. Haciendo el cálculo de tener 20 instancias “High-CPU Medium” levantadas 24/7 durante un año, el costo sería el siguiente:

TABLA IV: Costos de uso de nodos y tecnología AMAZON

Pago único anual:	$455
Horas utilizadas al año por instancia:	8760 (considerando uso 24/7)
Número de instancias:	20
Horas utilizadas al año total:	175 200
Costo por nodo por hora, esquema “Reserved Instances”:	$0,06
COSTO TOTAL CLÚSTER:	$10512

CONCLUSIONES Y RECOMENDACIONES

Conclusiones.

1.- El uso de la plataforma Hadoop utilizando Pig como un lenguaje generador de trabajos MapReduce permite el realizar consultas y análisis de información sobre volúmenes de información que no eran posible procesarse con un esquema tradicional de RDBMS.

2.- Esta tecnología no implica un reemplazo de una RDBMS tradicional, es importante entender que procesos o consultas de información que necesiten ser realizadas en línea y que no necesiten el procesar grandes volúmenes de información histórica deben seguir siendo realizadas por la RDBMS.

3.- La herramienta Pig al ser un lenguaje simple de procesamiento de flujo de datos, minimiza el tiempo requerido entre la formulación de un requerimiento ad-hoc y la codificación del mismo.

4.- El costo monetario de almacenar la información y procesarla en un clúster en la nube es dos órdenes de magnitud inferiores al de una solución similar utilizando una base de datos comercial.

5.- El uso de la plataforma Hadoop implementando MapReduce como una solución de computación distribuida, reduce al mínimo las dificultades y complejidades de programar aplicaciones distribuidas.

Recomendaciones:

1.- El sistema desarrollado asume que el clúster EC2 se encuentra previamente levantado antes de enviarse los requerimientos. A futuro puede mejorarse el sistema para que levante en demanda el clúster con una cantidad de nodos definida por el usuario.

2.- Amazon provee de una herramienta de almacenamiento alternativa a S3 denominada Elastic Block Store (EBS). La diferencia entre ambas, es que EBS permite acceso a nivel de bloque, de tal manera que este repositorio puede ser montado sobre el sistema operativo de los nodos de EC2. Por esto, provee un nivel de rendimiento mucho mayor al del acceso a repositorios S3 (los cuales utilizan Web services). En la actualidad, el acceso a EBS mediante Hadoop se encuentra en un período de Beta cerrada. [12]

3.- Los scripts utilizados para la automatización de la puesta en marcha de clústeres con máquinas virtuales con Hadoop y Pig, actualmente utilizan Hadoop 0.18. En esa versión de Hadoop aún no está disponible el acceso directo desde Hadoop/Pig hacia repositorios en S3, por lo cual el sistema desarrollado actualmente debe hacer el paso intermedio de extraer la información mediante la herramienta s3tools. En la versión Hadoop 0.20, desde Pig ya es posible referenciar directamente carpetas y archivos ubicados en S3, haciendo innecesario el paso intermedio de la transferencia de archivos.

4.- Aunque se utilice un esquema 24/7 con una gran cantidad de nodos reservados, siempre es preferible utilizar un número óptimo de nodos para cada tarea para no saturar el servicio.

5.- El sistema tal como se lo ha descrito, no tiene ningún control o priorización de trabajos enviados a procesar. Se recomienda integrar este sistema con el servicio Simple Queue System de Amazon, el cual provee de un esquema de envío de mensajes entre aplicaciones Web [13].

6.- Las aplicaciones posibles de este sistema son variadas, como la realización de minería de datos con fines de marketing [14], análisis de patrones de comportamiento inusuales en clientes para detectar fraude [15], entre otros.

REFERENCIAS BIBLIOGRÁFICAS

[1] **DERKS WIJNAND, DIJKSTRA SIETSE, JONKER WILLEM, WIJNANDS JEROEN,** Assessment of Scaleable Database Architectures for CDR Analysis - An Experimental Approach, KPN Research, The Netherlands

[2] **GROSSMAN, ROBERT L.,** A review of some analytic architectures for high volume transaction systems. In The 5th International Workshop on Data Mining Standards, Services and Platforms (DM-SSP '07), ACM, 2007.

[3] **DEAN, J. Y GHEMAWAT, S**. “MapReduce: Simplified Data Processing on Large Clusters“. En Memorias del Sixth Symposium on Operating System Design and Implementation (OSDI 2004). San Francisco, CA-EE.UU., Diciembre, 2004.

[4] **THE APACHE SOFTWARE FOUNDATION**. “Apache Hadoop”, < http://hadoop.apache.org > [Consulta: 20 de diciembre de 2010].

[5] **OLSTON, C., REED, B., SRIVASTAVA, U., KUMAR, R., AND TOMKINS, A**., Pig Latin: A not-so-foreign language for data processing. In International Conference on Management of Data (Industrial Track) (SIGMOD) , 2008.

[6] **WHITE, TOM,** Hadoop: The Definitive Guide. O’Reilly Media, Inc., June 2009, 83p.

[7] **PREPAID VOICE CDRS - PREPAID 4.5.1,** Tecnomen Corporation 2008

[8] **“PIG API – CLASS PIGSTORAGE”**, Apache Foundation, <http://hadoop.apache.org/pig/javadoc/docs/api/org/apache/pig/builtin/PigStorage.html>, [Consulta: 20 de diciembre de 2010].

[9] **“PIG API – CLASS PIGDUMP”**, Apache Foundation, <http://hadoop.apache.org/pig/javadoc/docs/api/org/apache/pig/builtin/PigDump.html>, [Consulta: 20 de diciembre de 2010].

[10] **“AMAZON SIMPLE STORAGE SERVICE (AMAZON S3)”**, Amazon Inc., <http://aws.amazon.com/s3/#pricing>, [Consulta: 20 de diciembre de 2010].

[11] **“AMAZON ELASTIC COMPUTE CLOUD (AMAZON EC2)”**, Amazon Inc., <http://aws.amazon.com/ec2/#pricing>, [Consulta: 20 de diciembre de 2010].

[12] **“AMAZON LAUNCHES EBS – PERSISTANT STORAGE FOR EC2”**, TechCrunchIT, <http://www.techcrunchit.com/2008/08/21/amazon-launches-ebs-persistant-storage-for-ec2>, [Consulta: 20 de diciembre de 2010].

[13] **“AMAZON SIMPLE QUEUE SYSTEM”,** Amazon Inc., http://aws.amazon.com/sqs/, [Consulta: 20 de diciembre de 2010].

[14] **CHEN, Q., HSU, M., DAYAL, U**., A Data-Warehouse / OLAP Framework for Scalable Telecommunication Tandem Traffic Analysis,

In Proc. of the 16th Intl. Conf. on Data Engineering (ICDE), IEEE CS Press, San Diego, CA, Mar. 2000.

[15] **SAHARON ROSSET, UZI MURAD, EINAT NEUMANN, YIZHAK IDAN, GADI PINKAS**, Discovery of Fraud Rules for Telecommunications: Challenges and Solutions, Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,