Análisis e implementación de una Entidad Externa De Mensajes Cortos (ESME) que opera bajo el Protocolo De Mensajería Escrita Punto A Punto (SMPP) versión 3.4 en Linux para el envío y recepción automático de Mensajes Escritos Cortos (SMS'S)

Giancarlo Omar Llerena Chévez [(1)], Romario Renee Pinda Paucar [(2)], MsC. José Miguel Menéndez [(3)]
Facultad de Ingeniería en Electricidad y Computación.
Escuela Superior Politécnica del Litoral (ESPOL)
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863 Guayaquil-Ecuador
gianller@espol.edu.ec [(1)], rrpinda@espol.edu.ec [(2)], jmenende@fiec.espol.edu.ec [(3)]

Resumen

El presente proyecto fue realizado con la finalidad de explicar las características del protocolo SMPP (Protocolo De Mensajería Escrita Punto A Punto), así como también los comandos y parámetros que este necesita para su funcionalidad. Para entender mejor este protocolo, se implementó un ESME (Unidad Externa de Mensajes Cortos) que interactuará con una SMSC (Central de Servicios de Mensajes Cortos) que también debe implementar el protocolo SMPP para que entre ellos se puedan comunicar. Este proyecto se lo desarrolló utilizando software de código abierto que evitará el pago de licencias y la dependencia directa de empresas para la realización de tareas de mantenimiento.

El ESME responderá solicitudes de los usuarios de manera automática a través de la SMSC. Estas solicitudes son palabras claves que envían los usuarios móviles, y que posteriormente el ESME escogerá y enviará un contenido asociado a esta palabra como respuesta a esta solicitud. Toda esta información se encuentra almacenada en una base de datos relacional que mejora la comprensión, garantiza la integridad y evita la duplicidad de los datos.

Palabras Claves: *SMPP, SMSC, ESME, implementación, usuarios móviles, operadora de contenido.*

Abstract

The project was made with the objective to explain the characteristics of protocol SMPP and also to show the parameters and the commands on how this works, for a better understanding of this protocol, it is going to be implanted the ESME which will be interacting with an SMSC, for this must implanted the protocol SMMPP so they can communicate between them. This project was developed using open source software to avoid paying licensing and reporting directly to companies for performing maintenance.

The ESME will answer to the requests of the users in an automatic way through the SMSC. These requests are key words that send the mobile users and after this the ESME will pick it up and send the content associated to the key word as a result of this request. All this information is stored in a relational database that improves understanding, it ensures the integrity and avoids duplication of data.

Keywords: *SMPP, SMSC, ESME, implementation, mobile users, operator content.*

1. Introducción

Hoy en día gracias al avance de la tecnología móvil y la creación de diversas aplicaciones es posible transferir datos y archivos multimedia de gran tamaño, de una manera rápida y a un menor costo.

Por este motivo, generalmente los usuarios prefieren usar las redes sociales para poder comunicarse y no el servicio de mensajería corta conocidos como SMS debido a la limitada cantidad de caracteres que puede contener y su alto costo; sin embargo, los mensajes cortos no dejan de ser útiles, seguros y rentables para las operadoras.

Otro uso común de los mensajes de texto cortos es el envío de datos a centros externos donde se podría obtener información de diversa índole, por ejemplo política, deportes, entretenimiento o noticias, esto depende a la organización al cual se está solicitando la información.

A estos centros externos donde procesan los mensajes cortos de los usuarios móviles se los denominan ESME conocidos en español como entidad externa de mensajes cortos que, haciendo uso del protocolo de SMPP que significa protocolo de mensajería escrita punto a punto, hacen posible la transferencia confiable y eficiente de mensajes cortos de textos.

2. Objetivos específicos.

El presente proyecto tiene como objetivo general implementar un ESME usando el protocolo SMPP a través del modelo de red TCP/IP. Para cumplir con este propósito se efectuaron los siguientes objetivos específicos:

1.- Implementar una interfaz gráfica usando el lenguaje de programación PHP en el lado del servidor conjuntamente con HTML y JAVASCRIPT en el lado del cliente para la administración del ESME de forma remota a través de un explorador web.

2.- Implementar una base de datos en MYSQL para almacenar los CDR's (registros detallados de llamadas) generados a partir de los mensajes que recibe y emite el ESME.

3.- Almacenar los comandos y mensajes SMPP que se usan para transmitir los mensajes cortos entre la SMSC y el ESME en archivos de logs para que puedan ser accedidos por los usuarios que administran el ESME.

3. Fundamentos Teóricos.

3.1 Descripción del Protocolo SMPP

En esta sección se detalla la implementación de protocolo SMPP.

SMPP es un protocolo que se usa para la transferencia de mensajes cortos desde un ESME hacia una SMSC, en esta transferencia el ESME se comporta como cliente mientras que la SMSC se comporta como servidor. Ambos elementos deben tener implementado el protocolo SMPP para que entre ellos puedan comunicarse.

3.2 Estructura general de los datos de protocolo SMPP.

SMPP PDU				
PDU Header (mandatory)				**PDU Body (Optional)**
command length	*command id*	*command status*	*sequence number*	*PDU Body*
4 octets	Length = (Command Length value - 4) octets			

Figura 3.1 Estructura de un mensaje SMPP.

Inmediatamente se describirán cada uno de los parámetros de cabecera:

Command_length
Define la longitud en octetos del mensaje SMPP. Esta longitud considera la cabecera, el cuerpo del mensaje e inclusive el mismo parámetro *command_length.*

Command_id
Este comando identifica el tipo de mensaje SMPP que se está transmitiendo. Por ejemplo pueden representar el comando *submit_sm*, *deliver_sm*, *deliver_sm_resp* entre otros.

Command_status
Indica el éxito o el fracaso de una solicitud SMPP. Es relevante en los mensajes SMPP de respuesta, mientras que en los mensajes de solicitud se coloca un valor NULL.

Sequence_number
Este parámetro contiene el número de secuencia y se utiliza para correlacionar la respuesta de un PDU SMPP con una solicitud. Este número puede oscilar de 0x00000001 y 0x7FFFFFFF.

3.3 Definición de los PDU's SMPP.

Se describe algunos de los PDU's que se usan para la comunicación de este protocolo.

PDU Bind
El propósito de esta operación es registrar una instancia de la SMSC con el ESME y solicitar una sesión SMPP (puede ser *transceiver*, *transmitter* o *receiver*) a través de una conexión de red para la entrega y recepción de mensajes.
PDU Unbind
El objetivo de este mensaje es desvincular la sesión que existe entre el ESME y la SMSC. Es decir el ESME informa a la SMSC que ya no desea ni enviar ni recibir mensajes.

PDU submit_sm
Esta operación solo la realiza el ESME y es donde está contenido el mensaje corto de texto. La SMSC envía el correspondiente *submit_sm_resp* para indicar al ESME que recibió correctamente el mensaje.

PDU *deliver_sm*
Este mensaje es emitido por la SMSC para que este pueda enviar mensajes al ESME. Para que la SMSC pueda enviar este mensaje, debe existir un enlace como *transceiver* o *receiver* entre ambas entidades.

PDU query_sm
Este mensaje es emitido por el ESME para consultar el estado de un mensaje corto enviado anteriormente.

3.4 Parámetros mandatorios.

A continuación se describe algunos parámetros necesarios para poder iniciar una sesión SMPP.

System_id
Este parámetro sirve para identificar un ESME o un SMSC durante la vinculación. Un *system_id* ESME identifica el ESME al SMSC.

Password
Este parámetro es usado por el SMSC en conjunto con el *system_id* para identificar también el ESME.

System_type
El parámetro *system_type* se usa para categorizar el tipo de ESME que es vinculante para el SMSC.

Addr_ton, source_addr_ton, dest_addr_ton, esme_addr_ton
Estos parámetros definen el tipo de número (TON) para ser utilizados en los parámetros de dirección del ESME.

TON	Valor
Desconocido	0
Internacional	1
Nacional	10
Red especifica	11
Numero de abonado	100
alfanumérico	101
Abreviado	110
Todos los demás valores reservados.	

Figura 3.2 Valores que se atribuyen al parámetro TON

Addr_npi, source_addr_npi, dest_addr_npi, esme_addr_npi
Estos campos definen el indicador plan numérico conocido como NPI para ser utilizado en los parámetros de dirección de los ESME.

NPI	Valor
Desconocida	0
ISDN (E163/E164)	1
Data (X.121)	11
Telex (F.69)	100
Land Mobile (E.212)	110
Nacional.	1000
Privada	1001
ERMES	1010
Internet (IP)	1110
WAP Client Id (to be defined by WAP Forum)	10010
Todos los demás valores reservados.	

Figura 3.3 Valores que atribuyen a NPI.

3.5 Sesiones SMPP.

Para poder iniciar una sesión en este protocolo se debe especificar como se va hacer la conexión, la cual puede ser en sesión *transmitter*, *receiver*, *transceiver*. Se detallara cada sesión a continuación:

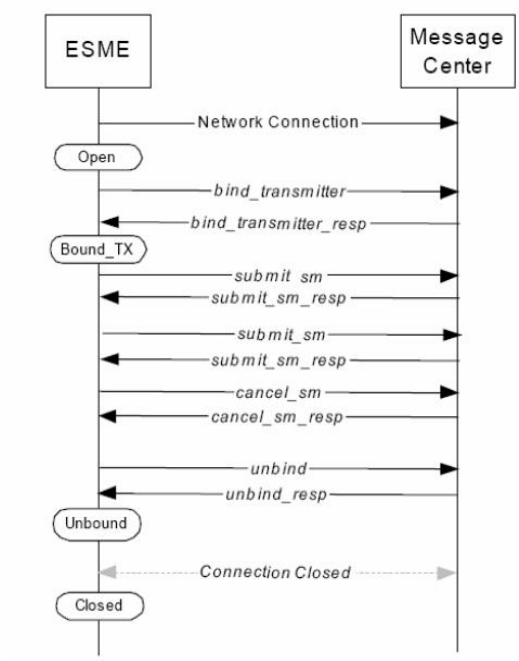

Figura 3.4 Sesión *Transmitter*.

En la figura 3.4 se detalla la sesión *transmitter*, esta sesión es unidireccional, primero se establece una conexión entre el ESME y el SMSC, luego se registra o se levanta una sesión *transmitter* con el parámetro PDU *bind_transmitter*, el ESME le envía un PDU *submit_sm* a el SMSC, la cual contendrá el mensaje de texto corto, el SMSC le responderá con un PDU *submit_resp* para verificar si el mensaje llego o no de forma correcta. Para finalizar la sesión *transmitter* se hace una desconexión con el parámetro *unbind* y finalmente se cierra la sesión.

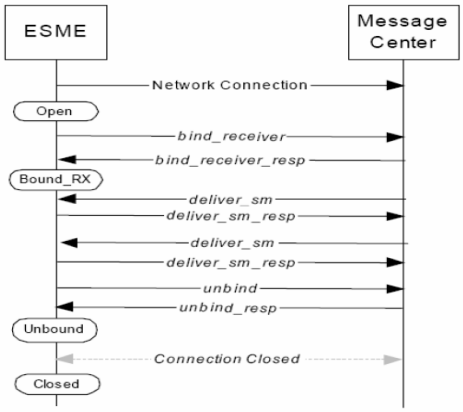

Figura 3.5 Sesión *Receiver*.

En la figura 3.5 se detalla la sesión *Receiver*, esta sesión consta de tres partes como lo son la conexión en estado de sesión receiver, los mensajes propios de esta sesión, y la desconexión de la sesión receiver.

Primero se estable una conexión entre el ESME y el SMSC, se registra una instancia o se levanta una sesión receiver, el SMSC le envía un mensaje de texto por el PDU *deliver_sm*, y el ESME le responde con un PDU *deliver_sm_resp* a el SMSC, para verificar si el mensaje llego o no de forma correcta.

Para finalmente desconectar la sesión y cerrar la misma.

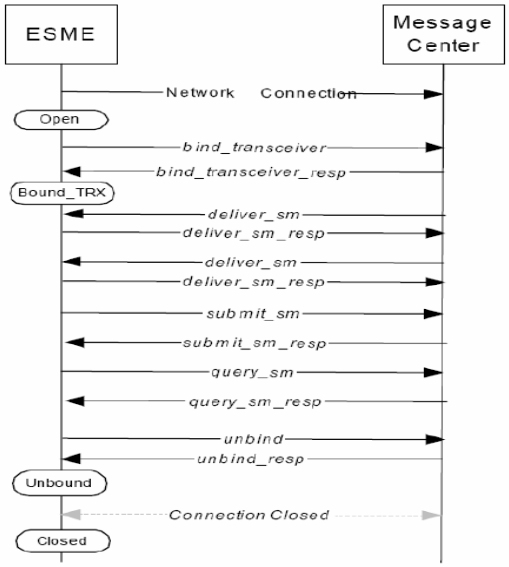

Figura 3.6 Sesión *transceiver*.

Se muestra el inicio de sesión como *transceiver* en el cual al igual que los dos casos anteriores, debe existir un inicio de sesión. Pueden operar como una sesión de mensajes dúplex el SMSC y ESME, es decir los mensajes se intercambian en ambas direcciones.

3.6 Funcionamiento general del protocolo SMPP entre SMSC y ESME.

En la Figura 3.7, se muestra gráficamente el proceso normal que se debe seguir para que el ESME pueda responder y procesar la respuesta a una petición que se origina de la SMSC. Como ya se mencionó, una de las limitaciones que se presentan en este proyecto es que no habrá teléfono móvil verdadero que envíe un mensaje a la SMSC sino que se utilizará otro ESME que pueda simular el dispositivo; sin embargo, se muestra a este ESME como el dispositivo celular.

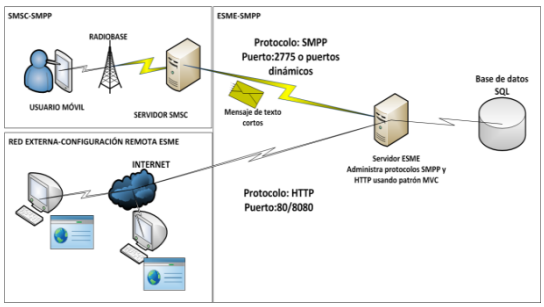

Figura 3.7 Vista general del funcionamiento entre ESME, SMSC.

En la Figura 3.8, se muestran los procesos que se deben seguir para que el usuario móvil pueda receptar el mensaje que solicitó a partir de una palabra clave que él mismo envió. Solamente se muestra el proceso en donde interactúa el protocolo SMPP, más no el proceso para configurar el ESME a través de la aplicación web.

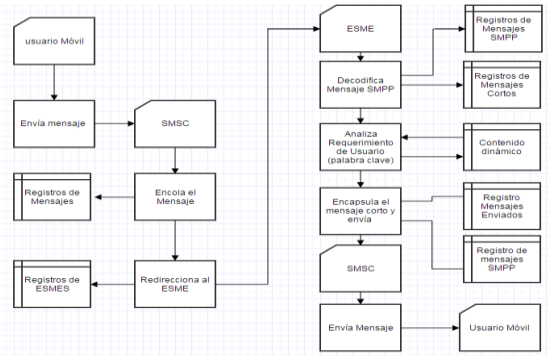

Figura 3.8 Diagrama de flujo de datos que representa el envío de mensajes desde el ESME al usuario móvil a partir de un requerimiento.

4. Desarrollo del proyecto.

4.1 Breve descripción de la solución.

En esta sección se explicará de manera detallada todas aquellas herramientas que se usaron para la implementación del ESME, y también aquellas aplicaciones, librerías, y patrones de diseño para el desarrollo de la aplicación web. Para tener una mejor comprensión del desarrollo de esta solución se llevará un orden específico descrito de la siguiente manera:

1. Configurando el entorno de desarrollo: En esta sección se describe de manera breve el sistema operativo que se usó y los componentes que se necesitan para acondicionar el entorno de desarrollo y posteriormente implementar la aplicación.

2. Herramientas para la creación de la aplicación y base de datos: Aquí se muestran cuáles son las herramientas que se necesitan para desarrollar la aplicación web que permitirá la administración del ESME y la base de datos.

3. Arquitectura y marcos de trabajo de la aplicación: En esta parte se detallan cuáles son los patrones que se usaron para el diseño de la aplicación web así como también una breve descripción de los marcos de trabajo que se usaron tanto del cliente como del servidor para que el desarrollo de la misma no se torne compleja.

4. Gestor de base de datos y modelo entidad relación: En esta parte se define lo que es un modelo entidad relación y qué gestor de base de datos se usó para el almacenamiento de la información.

5. Funcionamiento de la aplicación: Se describirá de manera detallada cuales son los pasos que el usuario debe seguir para el control del ESME a través de una aplicación web.

4.2 Configurando el entorno de desarrollo.

Para instalar la aplicación se necesita: una computadora y un servidor web en donde se coloca en el directorio raíz el proyecto para poder acceder a ella mediante un explorador web.

Figura 4.1 sistema operativo Linux

Como se está trabajando con herramientas código abierto, el sistema operativo que se usó es Ubuntu Server que básicamente es una distribución de Linux.

Figura 4.2 Servidor Apache.

Apache es un servidor web de código abierto diseñado específicamente para transferir datos de hipertexto, por ejemplo páginas web a través del protocolo HTTP.

Figura 4.3 Logo de lenguaje de programación PHP.

Como la aplicación que controlará al ESME es una aplicación web, se eligió a PHP como lenguaje de programación ya que este lenguaje trabaja en el lado del servidor

Figura 4.4 Logo del gestor de base de datos MYSQL.

Para que se puedan almacenar los datos, por ejemplo los mensajes enviados o recibidos, y también los comandos y parámetros que se usan para controlar al ESME se instaló en el servidor MYSQL que es un sistema gestor de base de datos relacional

4.3 Gestor de base de datos y modelo entidad relación.

Para explicar lo que se realizo en esta sección se uso el lenguaje SQL que se traduce como el lenguaje de consulta estructurado, soporta la mayoría de los motores de base de datos robustos como SQL SERVER, ORACLE o MYSQL.

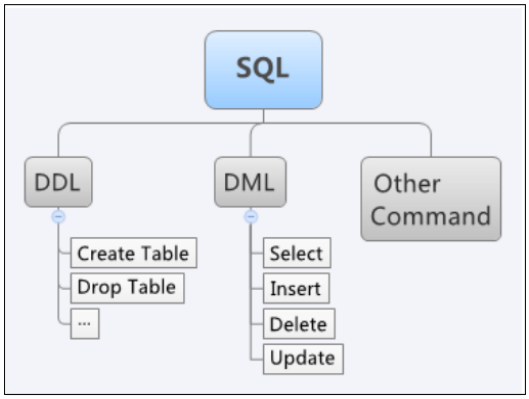

Figura 4.5 Lenguaje SQL para el manejo de base *de* datos.

Se muestran algunos de los comandos SQL que se pueden ejecutar en la base de datos. Para el manejo de las consultas, actualización, agregación, y eliminación de los datos se utilizó el motor de base de datos MYSQL, que soporta y ejecuta el lenguaje SQL.

5. Análisis de resultados.

Se muestra una captura por wireshark en donde se exponen los datos que se transmiten desde el servidor al cliente y viceversa. Cabe mencionar que esta gráfica se la capturó cuando se integró el ESME desarrollado en el presente proyecto y la SMSC desarrollada en la tesis "Análisis e implementación de un servidor de Protocolo de Mensajería Escrita Punto a Punto (SMPP) versión 3.4 en Linux que interactúe con un cliente SMPP en el envío y recepción de Mensajes Cortos (SMS's) y que genere archivos de Registros de Datos de Llamadas, CDRs". En ella se puede observar que se muestran algunos de los parámetros que se configuraron en la aplicación web y que wireshark los interpreta fácilmente ya que esta aplicación de red toma en cuenta la implementación del protocolo SMPP.

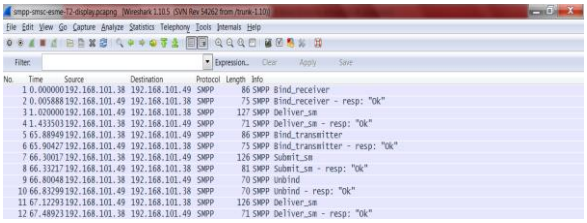

Figura 5.1 Captura en wireshark de un paquete SMPP.

5.1 Funcionamiento del ESME.

CODEIGNITER se basa en un patrón de diseño como MVC (Patrón de arquitectura de software Modelo-Vista-Controlador) que en significa modelo, vista y controlador. Este patrón de diseño de software permite separar la lógica del negocio con la interfaz del usuario, gráficamente se determina en la Figura 5.2.

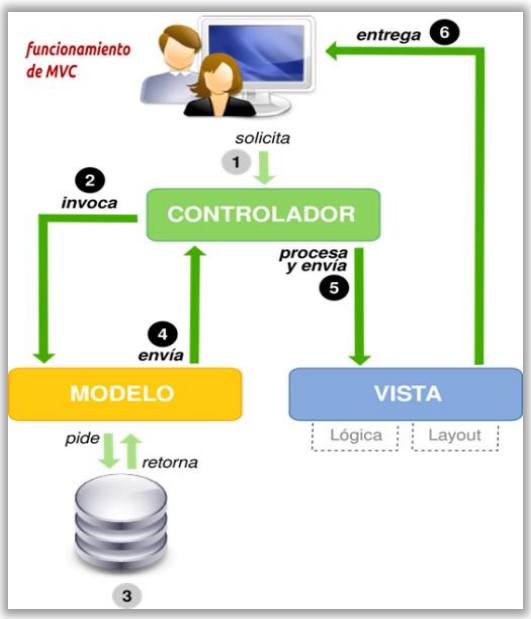

Figura 5.2 Patrón de desarrollo de software MVC

CSS (Hojas de estilo en cascada) permite insertar un formato personalizado a todas las etiquetas HTML permitiendo desarrollar una aplicación profesional y amigable. Para el diseño de la aplicación se usó el marco de trabajo o *framework* de CODEIGNITER basado en el lenguaje de programación de PHP

Figura 5.3 CODEIGNITER, Framework PHP

6. Conclusiones.

Gracias al desarrollo del presente proyecto se pudo cumplir con los objetivos propuestos y llegar a las siguientes conclusiones:

1. En el desarrollo del proyecto se utilizaron varias herramientas y conceptos tanto de informática como de telecomunicaciones. Gracias a esta integración, se pudo implementar una aplicación profesional, capaz de cubrir varias necesidades en el campo de las telecomunicaciones.

2. El uso de herramientas de código abierto para la implementación del ESME sirvió para no preocuparse por el pago de licencias tanto en el sistema operativo, el motor de base de datos y de los IDE's.

3. Conforme se estuvo desarrollando el proyecto se pudo constatar que la velocidad de comunicación entre la SMSC y el ESME, así como también el enlace para la administración remota de este último, depende del ancho de banda del enlace que poseen. En caso de implementar el ESME en redes públicas, se debe solicitar al proveedor de internet un ancho de banda que permita una conexión con una velocidad aceptable.

4. La versión 3.4 del protocolo SMPP fue suficiente para implementar un ESME robusto y eficiente, que tiene la capacidad de optimizar las conexiones con la SMSC gracias a la implementación de la sesión *transceiver*.

5. Se pudo percatar que el lenguaje SQL es compatible con el lenguaje de programación PHP. SQL permitió manejar los registros de la base de datos, y gracias a esta compatibilidad se pudo enviar los datos desde la base a la aplicación y viceversa fácilmente.

7. Recomendaciones.

Una vez finalizado el proyecto y después de haber explicado su funcionamiento se presentan a continuación las siguientes recomendaciones:

1. Antes de implementar cualquier solución es importante conocer e investigar cuáles son las tendencias actuales para el desarrollo de aplicaciones óptimas, para que los usuarios vean en esta solución una herramienta amigable y entretenida.

2. En caso de implementar el proyecto en un entorno corporativo y profesional, es recomendable instalar la aplicación web en un servidor robusto de alta capacidad, para brindar un mejor servicio y acceder a esta desde cualquier dispositivo conectado al internet.

3. Antes de comenzar a implementar cualquier protocolo de comunicación (no solamente el protocolo SMPP), es importante dominar cuales son los procesos y pasos que se deben seguir ya que, si no se tiene en claro estos procesos es posible que se termine desarrollando una aplicación que tenga un comportamiento no deseado.

4. Así mismo antes de implementar un protocolo de comunicación es importante estudiar cuales son las

versiones y especificaciones más recientes del mismo y si es posible implementar la aplicación tomando en cuenta la última versión o una versión estable. Actualmente el protocolo SMPP se encuentra en la versión 5.0, sin embargo se implementó la versión 3.4 que sigue siendo una versión muy usada y completamente estable.

8. Referencias.

[1] SMPP Developers Forum, http://www.smstrade.de/pdf/smpp.pdf, fecha de consulta noviembre del 2014.

[2] A.Olmos, http://es.scribd.com/doc/77539272/Smpp-Intro#scribd, fecha consulta noviembre 2014.

[3] Luz Marina Santos Jaimes, http://dialnet.unirioja.es/descarga/articulo/4272055.pdf, fecha consulta noviembre 2014.

[4] Tecno Fanático, http://www.tecnofanatico.com/no-todo-es-windows-y-mac-os-x-las-distribuciones-de-linux-mas-interesantes-del-2014/, fecha de consulta noviembre 2014.

[5] T.A.S Foundation, http://www.apache.org/, fecha de consulta noviembre 2014.

[6] Genbeta, http://www.genbeta.com/herramientas/apache-http-server-2-4-ya-disponible-con-mejoras-generales-de-rendimiento, fecha de consulta noviembre 2014.

[7] T.P.Group, http://php.net/downloads.php, fecha de consulta noviembre 2014.

[8] SolucionesPm, http://www.solucionespm.com/los-mejores-frameworks-de-php/, fecha de consulta noviembre 2014.

[9] Oracle, http://www.mysql.com/downloads/, fecha de consulta noviembre 2014.

[10] Genbeta, http://www.solucionespm.com/los-mejores-frameworks-de-php/, fecha de consulta noviembre 2014.

[11] T.E Foundation, https://eclipse.org/downloads/, fecha de consulta noviembre 2014.

[12] E. Foundation, https://eclipse.org/downloads/, fecha de consulta noviembre 2014.

[13] Oracle, http://www.mysql.com/products/workbench/, fecha de consulta noviembre 2014.

[14] Oracle Corporation, http://www.mysql.com/products/workbench/, fecha de consulta noviembre 2014.

[15] J.A.R Arteaga, https://sites.google.com/site/betoresendiz27/home/ instalaciondeunservidorlamplinux-apache-mysql-php, fecha de consulta noviembre 2014.

[16] Intercom, http://www.mailxmail.com/curso-php-mysql-aplicaciones-web-1/programacion-cliente-servidor, fecha de consulta noviembre 2014.

[17] T. Lea, http://www.tonylea.com/2011/intro-to-phonegap-com-build-native-mobile-apps-using-htmlcssjavascript/, fecha de consulta noviembre 2014

[18] Ellislab, http://www.codeigniter.com/download, fecha de consulta noviembre 2014.

[19] E. Martínez, http://www.tutorialesenvideo.net/codeigniter/, fecha de consulta noviembre 2014.

[20] E. Bahit, http://www.monografias.com/trabajos89/poo-y-mvc-php/poo-y-mvc-php2.shtml, fecha de consulta noviembre 2014.