SUBSECRETARIA DE EDUCACIÓN SUPERIOR
DIRECCIÓN GENERAL DE EDUCACIÓN SUPERIOR TECNOLÓGICA
INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

TRABAJO PROFESIONAL

COMO REQUISITO PARA OBTENER EL TITULO DE:

INGENIERO EN SISTEMAS COMPUTACIONALES

QUE PRESENTA:

ELI ALEJANDRO MORENO LÓPEZ

CON EL TEMA:

INTEGRACIÓN DE PROYECTOS DE SOFTWARE MEDIANTE EL USO DE PATRONES Y ARQUITECTURAS DE DESARROLLO.

MEDIANTE:

OPCIÓN I
(TESIS PROFESIONAL)

TUXTLA GUTIÉRREZ, CHIAPAS **JUNIO 2013**

Índice

Capítulo I: Introducción

El presente trabajo tiene como propósito servir como un marco de referencia para todas aquellas personas interesadas en dedicarse a la programación y diseño de aplicaciones web.

Para tal fin, se ha recurrido a la utilización de casos de estudio y la demostración con un sistema anteriormente desarrollado en donde se emplean patrones de diseño y técnicas avanzadas en el modo de desarrollo.

Conoceremos el empleo de máquinas virtuales para ejecutar las aplicaciones, con lo cual dejamos olvidada la parte de configurar un servidor HTTP local y simplemente nos concentraremos en el desarrollo y construcción del sistema web.

También se hablará del proceso de desarrollo rápido de aplicaciones o RAD (Rapid Application Development) y su puesta en práctica al desarrollar aplicaciones con el lenguaje de programación PHP.

1.1 Antecedentes

Ingeniería en Sistemas Computacionales es una carrera que forma parte de la oferta educativa del Instituto Tecnológico de Tuxtla Gutiérrez, esta ha ido evolucionando con el paso del tiempo para garantizar el egreso de profesionistas con un nivel alto de conocimientos en las diversas áreas a las que se puede dedicar un Ingeniero en Sistemas Computacionales (ISC).

Citando el objetivo de la carrera:

> Formar profesionistas líderes, analíticos, críticos y creativos, con visión estratégica y amplio sentido ético, capaces de diseñar, implementar y administrar infraestructura computacional para aportar soluciones innovadoras en beneficio de la sociedad, en un contexto global, multidisciplinario y sustentable.[1]

Además encontramos que en el perfil de egresado se encuentra mención a la siguiente:

> Desarrollar, implementar y administrar software de sistemas o de aplicación que cumpla con los estándares de calidad con el fin de apoyar la productividad y competitividad de las organizaciones.[2]

Teniendo en cuenta lo anterior y lo que se imparte en el plan de estudios, es sabido que resulta contraproducente conocer muchos temas de forma general y no exista la especialización esto imposibilita que los futuros ISC se puedan dedicar a una sola línea

[1] Extracto de la página web del ITTG el día 14 de Enero de 2013
[2] Extracto de la página web del ITTG el día 14 de Enero de 2013

como lo es el *Desarrollo de Software* o *Sistemas*, siendo que existen materias que resultan ser bastante teóricas y no aterrizan las teorías en un proyecto y/o prototipo demostrativo en donde se muestre lo aprendido en clase.

Dadas las circunstancias en que son educados los futuros Ingenieros en Sistemas Computacionales (en adelante ISC) y a la falta de elementos que ayuden a su buena formación como Desarrolladores de Software estos realizan su trabajo de una forma desordenada, llenando de archivos, funciones, clases incompletas, o simplemente programación funcional siguiendo su intuición como programador novato, dando como resultado algo a lo que muchas veces se le llama código espagueti y que con el tiempo será difícil de mantener y continuar desarrollando.

Muchos de estos proyectos son presentados cada año en el departamento de Ingeniería en Sistemas Computacionales, como proyectos de residencia o titulación integrada, están escritos sin seguir lineamientos de desarrollo eficiente, pruebas de seguridad y patrones de diseño. Esto ocurre porque al alumno próximo ISC o futuro desarrollador de sistemas no se le han enseñado técnicas de desarrollo avanzado, muchas veces solo se le ha enseñado a programar de una manera informal y sin las guías adecuadas.

1.2 Proyectos presentados.

A continuación se enumeran los proyectos que servirán como casos de estudio, se presenta una descripción de cada proyecto extraída del sitio web donde se encuentran publicados, se suministra también la dirección web para su posterior consulta.

Ilustración 1 Foro de propuestas de proyectos para titulación integral

Foro de propuestas de proyectos para titulación integral

http://seminario.proyectosisctectuxtla.com

Es una aplicación perteneciente al Departamento de Ingeniería en Sistemas Computacionales, cuya finalidad es informar a los alumnos que proyectos se encuentran registrados y puedan contribuir a su desarrollo.

Sistema de control interno de calificaciones de Lic. En Informática e Ing. Gestión Empresarial.
http://www.sicet.com.mx/sicic

Es una aplicación desarrollada por un grupo de alumnos de Lic. en Informática, ayuda a los docentes a registrar las calificaciones parciales de cada alumno de Lic. en Informática o Ing. en Gestión empresarial.

Ilustración 2 Sistema de control interno de calificaciones de Lic. En Informática e Ing. Gestión Empresarial

1.3 Planteamiento del problema

La mayoría de los Ingenieros en Sistemas Computacionales recién egresados de la carrera al dedicarse al desarrollo de sistemas se encuentran con complicaciones al momento de comenzar a contribuir en proyectos existentes debido al desconocimiento de cierto conjunto de reglas de diseño de software.

El desarrollar proyectos a largo plazo sin políticas y lineamientos de desarrollo implica trabajar de forma desordenada acumulando tareas y rutinas que antes ya han sido escritas, experimentando con conceptos nuevos o viejos, y muchas veces retrasando el avance del mismo.

Citando a los proyectos desarrollados para residencia o titulación integrada que son presentados cada año en la carrera para su evaluación y acreditación del alumno, encontramos proyectos de todo tipo desde software para PC, Juegos hasta aplicaciones web que proveen algún tipo de servicio.

Durante mucho tiempo se han descrito técnicas de desarrollo aplicadas a proyectos, lamentablemente no existe una formación o enseñanza adecuada de estas técnicas para los estudiantes de Ingeniería en Sistemas Computacionales, quienes al estar prestando su servicio social o residencia profesional se ven forzados a realizar desarrollos desorganizados a razón de que no llegan a conocer mejores técnicas de desarrollo de proyectos grandes y de larga duración.

Aterrizando esto, se hace referencia a materias que son impartidas como: Fundamentos de Desarrollo de Software, Planificación y modelado, Desarrollo de proyectos de software, aplicaciones de Internet, las cuales pertenecen al plan de estudios ISIC-2004-296. También tenemos las siguientes materias: Ingeniería de Software y Programación Web del plan de estudio ISIC-2010-224. Todas estas materias en conjunto están pensadas para ayudar a comprender el desarrollo de sistemas, sin embargo tienen un punto débil, pues ninguna aterriza en algún ejemplo práctico diseñado para demostrar todo lo aprendido en teoría y se han centrado más en pedir ideas para crear proyectos bien o mal hechos, omiten la idea de que con un proyecto probado y/o desarrollado es más fácil hacer demostraciones y probar lo que se está analizando con base en lo aprendido en clase.

1.4 Objetivos

1.4.1 Objetivo general

Lograr la organización en el desarrollo de proyectos de Software para la web mediante una arquitectura basada en el uso de Zend Framework, Vagrant y Git.

1.4.2 Objetivos específicos

- Usar de Zend Framework como base para desarrollo de Aplicaciones Web.
- Usar Git para llevar un control de cambios.
- Utilizar Vagrant gestionar máquinas virtuales.

1.5 Justificación

Para solventar la falta de conocimientos, la forma desorganizada en que trabajan muchos de los alumnos de ISC y así como la falta de ejemplos reales de aplicación de los conceptos estudiados. En esta propuesta se hace la demostración de cómo el trabajo de forma organizada y la utilización de lineamientos establecidos que comprenden la utilización de un Framework ayudan a realizar un trabajo más limpio.

Pero no solo se motiva el uso de un Framework, como Zend Framework, sino que también se incita al uso de un sistema de control de versiones con el cual se lleva el control de cambios realizados al código fuente donde inclusive se puede saber quién ha realizado algún cambio y para la distribución del código fuente entre los colaboradores se propone la utilización de un servicio como Github o Bitbucket.

Finalmente para garantizar el trabajo en equipo se recomienda la utilización de Vagrant con quien se pueden crear equipos virtuales idénticos para que así en un equipo de trabajo exista la homogeneidad sobre el lugar donde se realizan las pruebas y previsualizaciones del desarrollo.

Con todo esto se logra: (1) tener un control sobre las modificaciones al código fuente, (2) un trabajo en equipo en diferentes sistemas operativos pero con máquinas virtuales con las mismas características, (3) una mejor organización al aplicar el patrón de diseño HMVC de Zend Frawork, (4) así como la distribución del código fuente entre los colaboradores del proyecto.

1.6 Alcances y limitaciones

1.6.1 Alcances

- Se logra reducir el tiempo de desarrollo al usar Zend Framework.
- Al usar un Zend Framework en su forma modular se puede dividir el trabajo en tareas específicas de cada módulo.
- Con el control de versiones se consigue tener un control sobre los cambios realizados dentro del código fuente del proyecto.
- Con ayuda del control versiones y un servidor centralizado se consigue llegar a un trabajo colaborativo.
- Con el uso de Vagrant se logra que el desarrollo se pruebe en equipos con las mismas características sin importar el sistema operativo anfitrión.

1.6.2 Limitaciones

- Para el uso de Vagrant es necesario tener ciertas características de Hardware.
- La curva de aprendizaje puede ser prolongada.

1.7 Estado del Arte

A continuación se presentan algunos casos de estudio al trabajar de manera tradicional, desordenada y sin utilizar metodologías que ayuden a dar seguimiento al proyecto, tanto como en la escritura de su código fuente como en la utilización de estándares de desarrollo, como el uso de MVC y la modularidad.

Cada caso de estudio será acompañado de un análisis sobre la metodología con la que fue desarrollado el proyecto.

1.7.1 Caso de estudio: Foro de propuestas de proyectos para titulación integral.

Debido a que no se tiene acceso al sistema para hacer un diagnóstico más certero, sólo se realizaran estimaciones con respecto a tiempo y estilo de desarrollo.

El tiempo estimado es de más-menos 6 meses de desarrollo. El lenguaje que fue seleccionado para programar el sistema fue PHP, probablemente utilicen programación funcional para realizar cada proceso dentro del sistema.

En cuanto a la organización, se ha encontrado una carpeta de funciones en donde se encuentran todas las rutinas utilizadas dentro del sistema, entre estas se encuentra la rutina de acceso al sistema, salida del sistema, probablemente se encuentre la configuración de acceso a la base de datos, entre otros. También cuenta con otras carpetas en donde se almacenan archivos CSS, Imágenes y rutinas JavaScript.

Con una rápida inspección al código html generado, se puede saber la organización de carpetas y funciones.

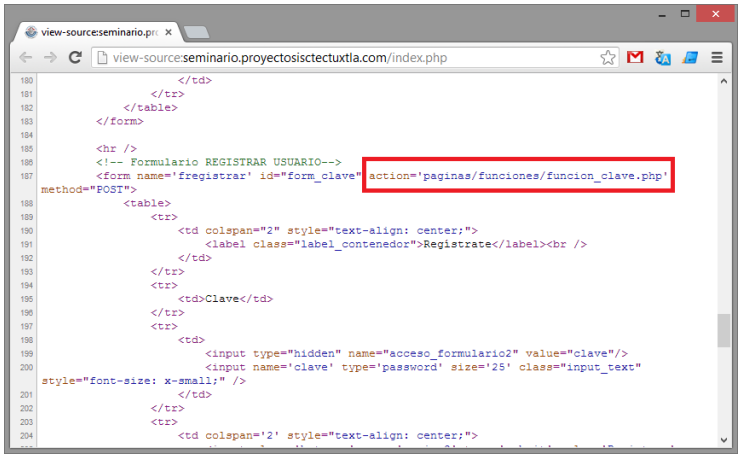


Ilustración 3 Inspección del código fuente

1.7.2 Caso de estudio: Sistema de control interno de calificaciones de Lic. En Informática e Ing. Gestión Empresarial.

Sistema desarrollado por alumnos de la carrera de Lic. en Informática en el periodo Agosto-Diciembre del año 2009, durante las prácticas de Residencia Profesional con una duración aproximada de 6. El sistema se encuentra escrito con lenguaje PHP y el estilo de programación es funcional.

Cuenta con diferentes páginas que se encargan de cumplir con funciones específicas dentro del sistema, como comprobar el acceso, realizar cambios, recuperación de contraseñas, etc.

Notas al respecto

Hay que denotar que a pesar de la poca experiencia de los estudiantes al realizar este tipo de proyectos han avanzado un pequeño tramo en su formación, pero sin la guía adecuada pueden no llegar a hacer aplicaciones que sean realmente competitivas, que se apeguen a patrones de diseño, que presten atención a la calidad de escritura de código y se preocupen también en la seguridad de su aplicación.

Los proyectos estudiados, han presentado el uso de técnicas de programación no recomendables para proyectos que deban de ser continuados por más tiempo, también presentan la utilización de funciones que no son aconsejadas para usar en la actualidad y serán eliminadas en versiones futuras de PHP, por lo que muchos de estos desarrollos tendrán problemas al intentar migrarlos a servidores actualizados.

1.7.3 Aplicación para el Laboratorio de Monitoreo Ambiental (LMA) de la Secretaría de Medio Ambiente e Historia Natural (SEMAHN).

La aplicación se le ha nombrado LMA y es accesible desde http://lma.tuxmapa.com.mx.

LMA ha sido desarrollado para agilizar la generación de documentos y organización de la información recolectada cada vez que se realiza un muestreo en base a solicitudes que llegan a la dependencia. Destaca la presentación de la información por medio de geolocalización (localización geográfica en base a coordenadas dentro de un mapa mundial), resumen de los estudios realizados, registro de clientes, registro de personal que podrá generar reportes, exportación de reportes como documento PDF.

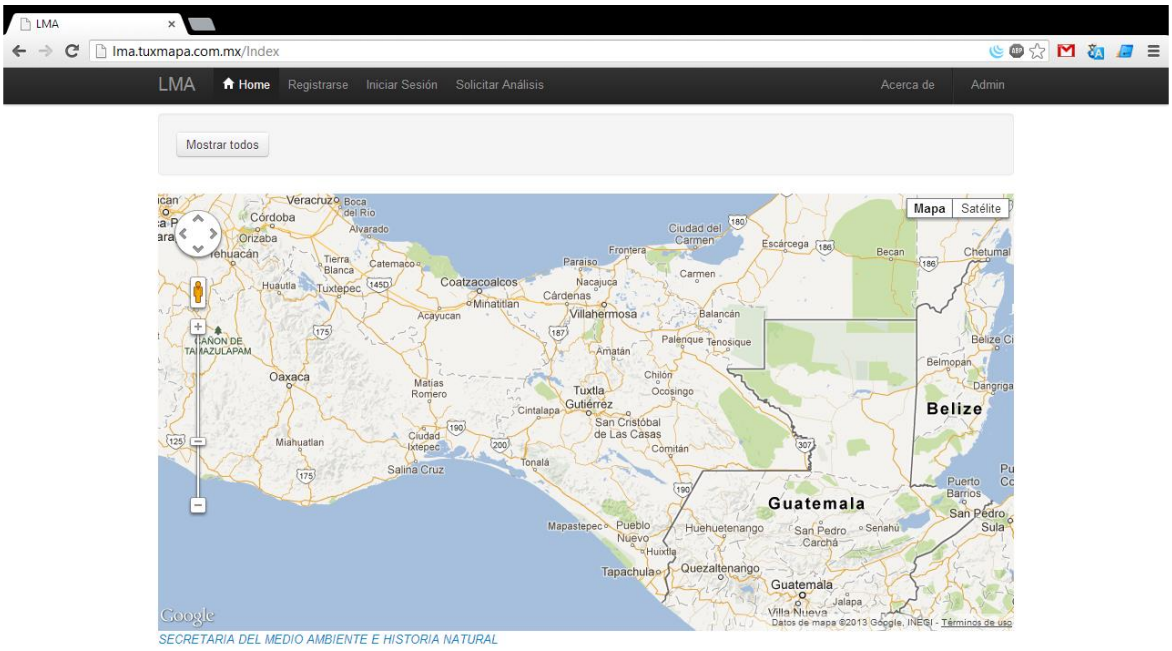


Ilustración 4 Aplicación de LMA

El tiempo de desarrollado del proyecto ha sido aproximadamente de 6 meses, se contempla el análisis de la información, planificación de la base de datos y la escritura del código fuente.

No se utilizó ningún sistema de control de versiones por lo que no se pueden conocer los estados anteriores del sistema, solo se conoce el estado final.

1.7.4 GeoAnúnciate

Es el nombre de una empresa pero también es el nombre de una aplicación web orientada a proporcionar información y localización ya sea de Servicios y/o Tiendas que se anuncien dentro del sitio.

Cuenta con un buscador que filtra palabras clave, una sección con filtro por categorías, muchos de los datos son alimentados por medio de Ajax. También utiliza la API de Google Maps para mostrar información dentro de mapa de la ciudad de Tuxtla Gutiérrez, Chiapas.

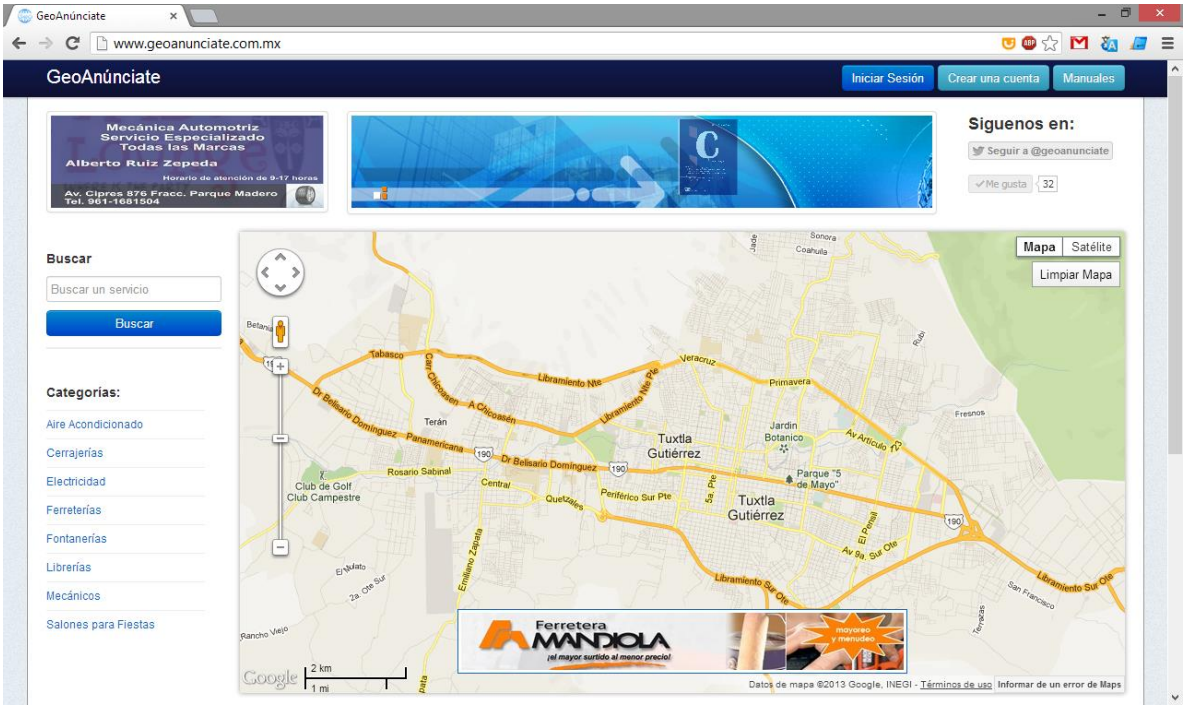


Ilustración 5 Página principal de GeoAnúnciate

El tiempo de desarrollo del sistema hasta su estado actual ha sido de aproximadamente un año lo cual comprende: análisis del nuevo esquema de trabajo, análisis y desarrollo de la base de datos, creación del código fuente del propio sistema.

Está construido sobre Zend Framework, por lo que el sistema tiene mejoras periódicas, en su última actualización se le ha dotado de una API (Application Programming Interface o Interfaz de Programación de Aplicaciones) con la cual la información contenida en GeoAnúnciate puede ser consultada por otras aplicaciones Web.

Cabe mencionar que el sistema cuenta con un Repositorio (Almacenamiento centralizado) de control de versiones, por lo que si se desea conocer y/o probar las versiones anteriores del sistema es posible visualizarlas. El sistema también brinda instantáneas o tags

(etiquetas) que son copias de lo que se encuentra en producción o mejor dicho se publicó para poderse usar.

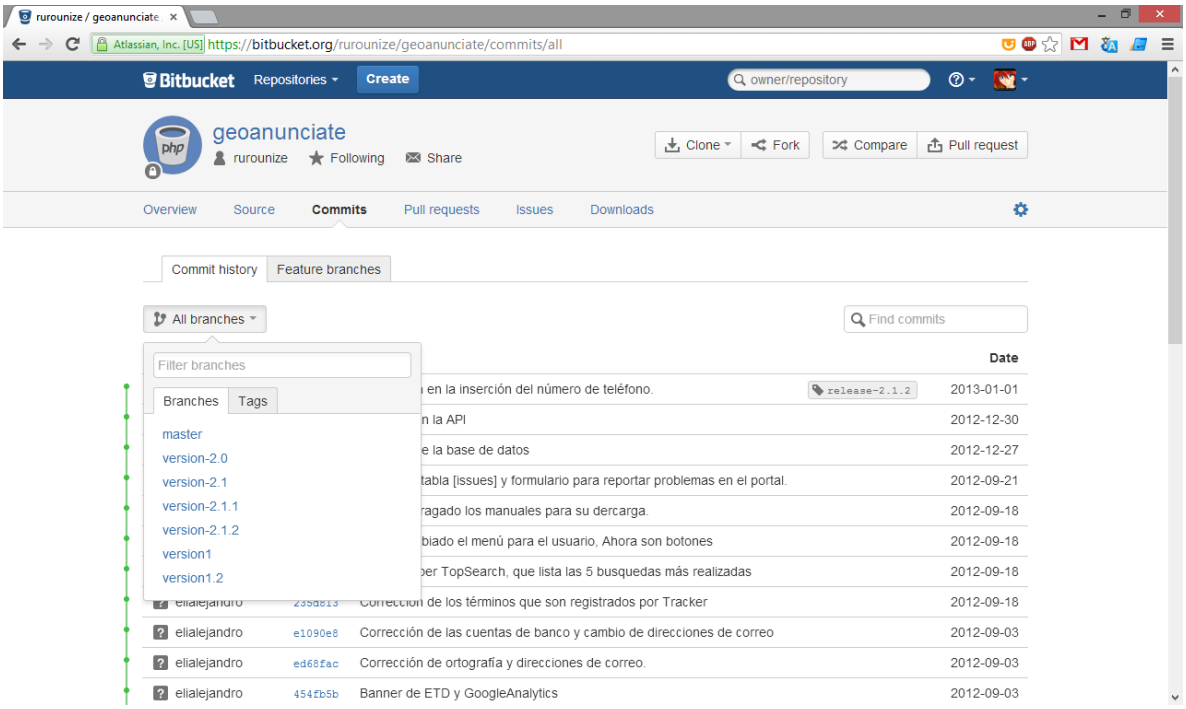


Ilustración 6 Repositorio en Bitbucket de GeoAnúnciate

Capítulo II: Marco teórico

2.1 Marco teórico técnico conceptual.

El marco teórico técnico está compuesto por las definiciones de palabras y/o frases clave que son mencionadas dentro del documento, además estas se encuentran organizadas en orden alfabético.

Apache HTTP Server: Es un servidor web HTTP de código abierto multiplataforma que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Su desarrollo comenzó en 1995 basándose inicialmente en el código de NCSA HTTPd 1.3. (Wikipedia, 2012)

Base de datos (BD): Es un conjunto de datos relacionados entre sí. Por **datos** entendemos hechos conocidos que pueden registrarse y que tienen significado implícito. (Elmasri & Navathe, 1997)

BitBucket: Es un sitio de alojamiento para los sistemas de control de versiones distribuidos (de sus siglas en Inglés DVCS, Ditributed Version Control Sistem) Git y Mercurial. La oferta de servicios incluye un programa de seguimiento (issue tracker) y wiki, así como la integración con una serie de servidores. (Atlassian, 2013)

CakePHP: Es un framework libre, de código abierto para el desarrollo rápido de aplicaciones para PHP. Es una estructura fundamental para ayudar a los programadores a crear aplicaciones web. (Cake Software Foundation, Inc. , 2013)

Cloud Computing: En español computación en la nube, concepto conocido también bajo los términos servicios en la nube, informática en la nube, nube de cómputo o nube de conceptos, es un paradigma que permite ofrecer servicios de computación a través de Internet.

Chef: Chef es un marco de integración de sistemas, construido para llevar los beneficios de la gestión de configuración de infraestructuras en la nube. No importa lo complejo, Chef hace que sea fácil de desplegar servidores y aplicaciones de gran escala a lo largo de toda la infraestructura. Con Chef, se escriben definiciones abstractas como el código fuente para describir como quiere cada parte de la infraestructura que se construirá, y luego aplicar esas descripciones para servidores individuales. Más información en: *http://wiki.opscode.com/display/chef/Home*.

Constante: Es un valor que no cambia durante la ejecución de un programa. Estos valores pueden ser decimales, enteros, hexadecimales, octales, etc.

Control de Versiones: El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante. (Chacon, 2009)

Codeigniter: Es un poderoso framework PHP con un tamaño muy reducido, construido para programadores PHP que necesiten una herramienta simple y elegante para crear aplicaciones web con todas las funciones. (EllisLab, 2013)

CVS: es sinónimo de "Concurrent Versioning System", y es un sistema de control de versiones diseñado para proyectos de software. El CVS puede tener varios usuarios simultáneamente en línea y trabajando en un mismo archivo. El papel de estos es hacer cambios en el código fuente (incluyendo corregir errores). Al mismo tiempo, las versiones anteriores se guardan y se pueden restaurar. (CVS HOME)

EC2: Servicio perteneciente a Amazon y significa *Elastic Compute Cloud*, el cual proporciona capacidad informática con tamaño modificable en la nube. Está diseñado para facilitar a los desarrolladores recursos informáticos escalables y basados en web. Más información en *http://aws.amazon.com/es/ec2/*. A grandes rasgos EC2 permite crear servidores de páginas web en pocos minutos.

EngineYard: Es un servicio PaaS (Plataform as a Service – Plataforma como servicio) empodera a los desarrolladores para planear, construir, desplegar y manejar aplicaciones en la nube. Proporciona un control sin igual y a elección, Engine Yard proporciona una nube confiable, una aplicación completa y soporte para los desarrolladores expertos y permite a las empresas centrarse en la creación de grandes aplicaciones, en lugar de la gestión de su plataforma. (Engine Yard, 2013)

Fila (Row): Es un con junto de datos extraídos de una tabla.

Framework: En el desarrollo de software, un framework (marco de trabajo), es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, con base a la cual otro proyecto de software puede ser más fácilmente organizado y desarrollado. (Wikipedia, 2012)

GIT: Nacido en 2005, es un Software de Control de Versiones Distribuido, diseñado por *Linus Torvalds*, siguiendo los objetivos que son: Velocidad, Diseño sencillo, Fuerte apoyo al desarrollo no lineal (miles de ramas paralelas), Completamente distribuido, Capaz de manejar grandes proyectos como el núcleo de Linux de manera eficiente (velocidad y tamaño de los datos). (Chacon, 2009)

GitHub: Es un servicio de almacenamiento de proyectos manejados con GIT, cuya finalidad es almacenar, compartir y trabajar de forma colaborativa en proyectos de código abierto.

Heroku: Proporciona una plataforma como servicio (PaaS) para construir, implementar y ejecutar aplicaciones de la nube usando Ruby, Node.js, Clojure, Java, Python y Scala. La arquitectura de esta plataforma incluye herramientas para la implementación y la administración, un tiempo de ejecución para la escalabilidad, tolerancia a fallos y un

sistema de add-ons para ampliar las capacidades de la plataforma. Más información en *http://www.heroku.com/*.

HMVC: Es una adaptación del MVC, el HMVC – Hierarchical-Model-View-Controller (Modelo-Vista-Controlador-Jerárquico) – paradigma pretende solucionar algunos de los problemas del MVC como gestión de datos, gestión de eventos y flujos de aplicaciones. HMVC ofrece una potente y fácil metodología de entender del diseño de capas para el desarrollo de una capa de presentación completa. (JavaWorld, 2000)

HTTP: Siglas de Hypertext Transfer Protocol (Protocolo de transferencia de Hipertexto) es un protocolo de nivel de aplicación para sistemas distribuidos y colaboración. Es genérico, protocolo sin estado, que se puede utilizar para muchas tareas más allá de su uso para hipertexto, como servidores de nombres y sistemas de administración de objetos distribuidos. (W3 org, 2002)

Lenguaje de programación: Es un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

MAMP: Es la abreviación de Macintosh, Apache, Mysql y PHP. Con solo unos clicks del mouse puedes instalar Apache, PHP y MySQL en una Mac OS X. (MAMP, 2013)

Máquina Virtual: Es la simulación de un equipo físico que permite ejecutar un Sistema Operativo sin modificaciones con todo el software instalado para ejecutarse en un ambiente especial, a esta máquina virtual se le conoce como *guest* o invitado, la cual se encuentra sobre un Sistema operativo existente (*host*). Esta máquina virtual es creada por un software de virtualización.

Modularidad: Es el diseño de un sistema que se divide en un conjunto de unidades funcionales (llamadas modulos) de las que puede estar compuesta una aplicación más grande. Un módulo representa una serie de problemas relacionados. Se puede incluir un conjunto de componentes relacionados, tales como características, vistas o lógica de negocio y trozos de infraestructura. Los módulos son independientes uno del otro, pero pueden comunicarse entre sí de una manera imprecisa. (Microsoft MSDN, 2013)

MVC: Es la abreviación de Model-View-Controller (Modelo-Vista-Controlador) es un patrón de diseño de software que separa la presentación, la lógica de negocio y el acceso a datos; con esta separación se logra un código mayor organizado, por lo que en proyectos grandes está separación es indispensable a la hora de codificar cuando más de un desarrollador está trabajando en la misma aplicación. (Zend, 2012)

MySQL: Es un sistema de gestión de bases de datos SQL Open Source, lo desarrolla, distribuye y soporta MySQL AB. El servidor de BD de MySQL es rápido, fiable y fácil de usar, es capaz de gestionar BD Relacionales en donde los datos se encuentran distribuidos en múltiples tablas. Es Open Source, lo que significa que es posible para cualquier persona utilizar y modificar el software para adaptarlo a sus necesidades. (MySQL, 2011)

Netbeans IDE: Es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender NetBeans IDE, es un producto libre y gratuito sin restricciones de uso. (Oracle Corporation and/or its affiliates., 2012)

Open Source: Es el término con el que se conoce al software distribuido y desarrollado libremente. El código abierto tiene un punto de vista más orientado a los beneficios prácticos de poder acceder al código, que a las cuestiones éticas y morales las cuales se destacan en el software libre. (Wikipedia, 2012)

PaaS: Plataform as a Service o Plataforma como servicio es una forma de apoyar el ciclo de vida completo de la entrega de aplicaciones y servicios web a través de la nube. PaaS combina una plataforma de desarrollo, recursos informáticos, la infraestructura y la implementación de servicios gestionados de hosting, por lo que puede reducir el costo, el tiempo y la complejidad del desarrollo de aplicaciones.

Patrón: Es la descripción de un problema que ocurre una y otra vez en nuestro entorno, para describir el núcleo de la solución a ese problema, de tal manera que la solución pueda ser usada más de un millón de veces más, sin tener que hacerlo de la misma manera dos veces. (Alexander, 1977)

PHP: Acrónimo de ***"PHP: Hypertext Preprocessor",*** es un lenguaje ***"Open Source"*** interpretado de alto nivel, especialmente pensado para desarrollos web y el cual puede ser incrustado en páginas HTML. La mayoría de su sintaxis es similar a C, Java y Perl y es fácil de aprender. La meta de este lenguaje es permitir escribir a los creadores de páginas web, páginas dinámicas de una manera rápida y fácil, aunque se pueda hacer mucho más con PHP. (The PHP Documentation Group, 2012)

Programación: Es el proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático. También se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos.

Programas: Un programa es un conjunto de instrucciones que una vez ejecutadas realizarán una o varias tareas en una computadora. Sin programas, estas máquinas no pueden funcionar correctamente. Al conjunto general de programas, se le denomina

software y así, se refiere al equipamiento lógico o soporte lógico de una computadora digital.

Puppet: Es un software de automatización que ayuda a los administradores de sistemas gestionar la infraestructura a lo largo de su ciclo de vida, desde el aprovisionamiento y la configuración para la gestión de parches y cumplimiento. Usando Puppet, se puede fácilmente automatizar tareas repetitivas, desplegar rápidamente aplicaciones críticas, y gestionar proactivamente el cambio, escalando desde 10 servidores hasta 1000, en la nube. Más información en *https://puppetlabs.com/puppet/what-is-puppet/*.

Rackspace Cloud: Es un servicio de computación en la nube, que se conoce como PaaS (Plataform as a Service en español Plataforma como servicio) donde se ofrecen equipos virtuales para montar servidores de aplicaciones web, donde sus principales características son: elasticidad y bajo costé. Más información en *http://www.rackspace.com/cloud/*.

Ruby: Un lenguaje de programación dinámico, orientado a objetos y de código abierto enfocado en la simplicidad y productividad. Su elegante sintaxis se siente natural al leerla y fácil al escribirla. (Ruby Lang Org., 2013)

Subversion: Es un sistema de control de versiones libre y de código fuente abierto. Es decir, Subversion maneja ficheros y directo- rios a través del tiempo. Hay un árbol de ficheros en un repositorio central. El repositorio es como un servidor de ficheros ordina- rio, excepto porque recuerda todos los cambios hechos a sus ficheros y directorios. Ésto le permite recuperar versiones antiguas de sus datos, o examinar el historial de cambios de los mismos. (Collins-Sussman, Fitzpatrick, & Pilato, 2004)

Tabla: Es el conjunto de datos correspondientes a un asunto que se encuentran ordenados, estructurados y disponibles para usarse.

Variable: Es aquella que contiene un valor la cual puede ser modificada a lo largo de la ejecución de la aplicación.

Virtualización: En informática es la capacidad de crear ambientes aislados de una máquina física que presentan todas las características de un equipo físico y pueden tener acceso controlado a algunas partes de hardware como USB, CD/DVD, FLOPPY, Sistemas de Ficheros compartidos, puertos Ethernet, e incluso al mismo microprocesador.

WampServer: Es un entorno de desarrollo web para Windows. Permite crear aplicaciones web con Apache 2, PHP y una base de datos MySQL. Además, cuenta con phpMyAdmin que permite manejar fácilmente las bases de datos. (Bourdon, 2013)

Zend Framework: Es un framework open source (código abierto) para el desarrollo de aplicaciones y servicios web con PHP 5. Zend Framework está implementado usando

código 100% orientado a objetos. La estructura de componentes de Zend Framework es algo única; cada componente está diseñado con poca dependencia en otros componentes. Esta arquitectura de bajo acoplado permite a los desarrolladores usar los componentes individualmente, este diseño es llamado “utilizar a voluntad”. (Zend Framework, 2012)

2.2 Marco teórico específico.

2.2.1 MVC (Modelo-Vista-Controlador)

Es uno de los patrones más citados (y más citados erróneamente). Comenzó como un *framework* desarrollado por Trygve Reenskaug para la plataforma Smalltalk en 1970. Desde entonces, ha desempañado un rol influyente en más frameworks de UI (User interface) y en el pensamiento acerca del diseño.

Como trabaja

El MVC considera tres papeles: *el modelo, el controlador y la vista*.

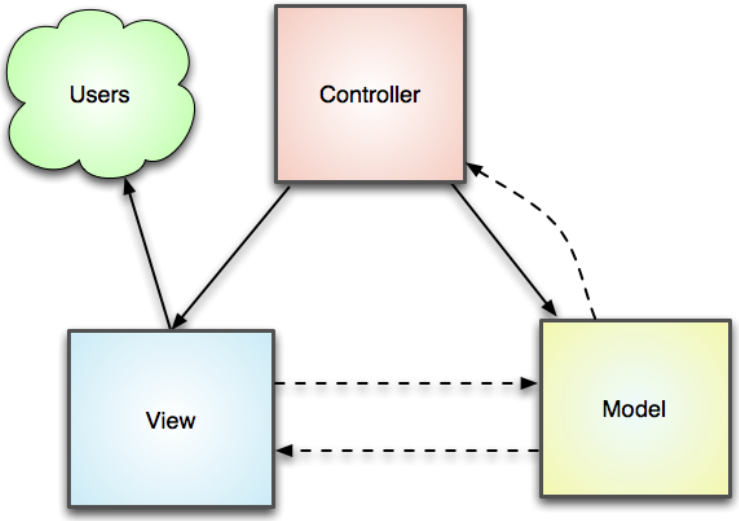


Ilustración 7Representación del funcionamiento del MVC[3]

- **Modelo:** Es un objeto que representa alguna información acerca del dominio. Este es un objeto no visual que contiene todos los datos y comportamientos distintos a los utilizados por la interfaz de usuario. En su forma más pura OO (Orientado a Objetos) el modelo es un objeto dentro del Modelo del Dominio.
- **Vista:** Define exactamente que es presentado al usuario. Normalmente los controladores pasan los datos a cada vista para ser representados en la vista con algún formato. Las vistas también recopilan datos del usuario, con formularios.
- **Controlador:** Los controladores integran el patrón siempre. Ellos manipulan a los modelos, decide que va a mostrarse basado en la petición del usuario y en otros factores, pasa los datos que cada vista necesita, o sede el control completo a otro controlador.

[3] Imagen tomada de: http://framework.zend.com/manual/1.12/en/learning.quickstart.intro.html, 8 de Noviembre de 2012

2.2.2 HMVC (Hierarchical-Model-View-Controller)

Como se mencionó en la parte del marco teórico técnico, el HMVC fue presentado en el año 2000 por Javaworld como una forma de dar solución a algunos de los problemas que se presentan al utilizar MVC en proyectos de desarrollo de gran tamaño.

Javaworld define al HMVC:
Como una adaptación al MVC, el paradigma HMVC pretende solucionar algunos de los problemas que no soluciona el MVC (gestión de datos, gestión de eventos y flujo de aplicaciones). Mediante el trabajo de campo se ha desarrollado este patrón, el cual ofrece un potente y fácil entender de la metodología de diseño por capas para el desarrollo de una capa de presentación completa. Mientras que el MVC proporciona un marco eficiente para el desarrollo de la interacción GUI, HMVC escala al nivel de cliente completo. Algunos de los beneficios clave de la responsabilidad basada en la arquitectura de capas incluyen:

- Definición de comunicación intracapas y aislamiento de capas más altas.
- Define la comunicación entre capas con acoplamiento mínimo.
- La localización de la exposición al código de terceros.

Algunos de los beneficios clave del uso de HMVC revelan los beneficios de la orientación a objetos. Una arquitectura en capas de manera óptima:

- Reduce las dependencias entre partes dispares del programa.
- Aliente a la reutilización de código, componentes y módulos.
- Aumenta la capacidad de mantenimiento mientras que facilita la extensibilidad.

Barry Cogan define al HMVC de la siguiente manera:

HMVC es una evolución del patrón MVC usado por muchas aplicaciones web hoy en día. Surgió como una respuesta a los problemas de escalabilidad aparentes dentro de las aplicaciones que utilizan el MVC. La solución fue presentada en el sitio de Javaworld, en Julio de 2000, propuesta para el estándar MVC convertido en capas en una “Jerarquía de padre-hijo con capas MVC (hierarchy of parent-child MCV layers)”. La siguiente imagen ilustra la forma en cómo trabaja:

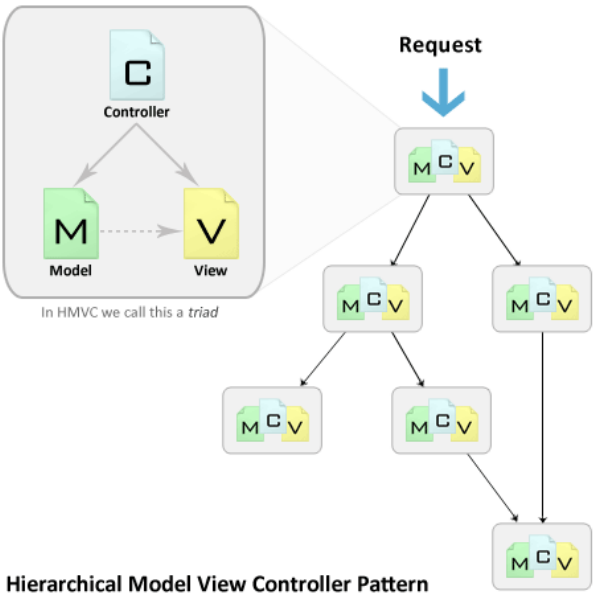


Ilustración 8 hierarchy of parent-child MCV layers[4]

2.2.3 Vagrant

Vagrant es una herramienta cuyo propósito es simplificar algunas de las tareas de los desarrolladores de aplicaciones web quienes necesitan instalar un servidor con todo lo necesario para ejecutar el proyecto en el que están trabajando. En pocas palabras es un servidor Linux virtual configurable con cualquier tecnología web que sea necesaria.

El sitio oficial del sistema, nos explica lo siguiente

> *¿Por qué **Vagrant**?*
> *Los desarrolladores web usan ambientes virtuales todos los días con sus aplicaciones web. Desde **EC2**, **Rackspace Cloud** hasta soluciones especializadas como **EngineYard** y **Heroku**, la virtualización es la herramienta elegida para una fácil implementación y gestión de infraestructura. Vagrant pretende dar esos mismos principios y ponerlos a trabajar en el corazón del ciclo de vida de la aplicación. Al ofrecer fácil configuración, máquinas virtuales ligeras, reproducibles y portátiles dirigidas a entornos de desarrollo, Vagrant ayuda a maximizar la productividad de usted y su equipo.*
>
> *Vagrant es una herramienta de desarrollo que se coloca en los hombros de gigantes, utilizando tecnologías probadas y comprobadas para lograr su magia. Vagrant utiliza Oracle VirtualBox para crear sus máquinas virtuales y luego usa **Chef** o **Puppet** a disposición ellos.*[5]

[4] Imagen tomada de http://net.tutsplus.com/tutorials/php/hvmc-an-introduction-and-application/, 14 de Noviembre de 2012
[5] Extracto de la web del proyecto: http://docs.vagrantup.com/v1/docs/getting-started/why.html, 12 de Enero 2013

Vagrant es capaz de manejar los siguientes sistemas operativos:

- OpenSuse
- CentOS
- Ubuntu
- Fedora
- OpenBSD
- FreeBSD

También es capaz de instalar automáticamente cualquiera de los siguientes servidores web:

- Apache2 + PHP
- Tomcat
- Ruby
- Ruby on Rails
- Phyton

Lista de comandos básicos con vagrant

Comando	**Descripción**
vagrant init *NOMBRE*	Inicia un nuevo servidor vagrant. *NOMBRE* distribución de Linux.
vagrant box add *NOMBRE URL o ARCHIVO.box*	Agrega una nueva máquina virtual preconfigurada para usar con vagrant. *NOMBRE* nombre corto de la máquina virtual, *URL* dirección de donde descargar la máquina virtual.
vagrant up	Inicia la máquina virtual y ejecuta la preconfiguración dada por el archivo Vagrantfile
vagrant halt	Apaga la máquina virtual
vagrant destroy	Destruye la máquina virtual

Requisitos para utilizar Vagrant

- Un equipo con por lo menos 2Gb de RAM y 10Gb de disco duro libre.
- Recomendable un procesador que soporte tecnologías de virtualización AMD64/Intel64.
- Tener instalado Ruby en su última versión.
  Se puede obtener de: http://rubyinstaller.org/downloads/ o http://www.ruby-lang.org/es/downloads/.
- Tener instalado Oracle VirtualBox.
  Se puede obtener de: https://www.virtualbox.org/.

La instalación de vagrant es muy sencilla basta obtener el instalador de su página oficial: http://www.vagrantup.com/.

2.2.4 Git (Control de Versiones descentralizado)

Acerca del control de versiones:

¿Qué es el control de versiones, y por qué debería importarte? El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante.

Si eres diseñador gráfico o web, y quieres mantener cada versión de una imagen o diseño (algo que sin duda quieres), un sistema de control de versiones (Version Control System o VCS en inglés) es una elección muy sabia. Te permite revertir archivos a un estado anterior, revertir el proyecto entero a un estado anterior, comparar cambios a lo largo del tiempo, ver quién modificó por última vez algo que puede estar causando un problema, quién introdujo un error y cuándo, y mucho más. Usar un VCS también significa generalmente que si fastidias o pierdes archivos, puedes recuperarlos fácilmente. Además, obtienes todos estos beneficios a un coste muy bajo.

Tipos de control de versiones:

Sistemas de control de versiones locales

El método de control de versiones usado por mucha gente es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron, si son atentos). Este enfoque es muy común porque es muy simple, pero también tremendamente propenso a errores. Es fácil olvidar en qué directorio te encuentras, y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.

Sistemas de control de versiones centralizados

El siguiente gran problema que se encuentra la gente es que necesitan colaborar con desarrolladores en otros sistemas. Para solventar este problema, se desarrollaron los sistemas de control de versiones centralizados (Centralized Version Control Systems o CVCSs en inglés). Estos sistemas, como *CVS* y *Subversion*, tienen un único servidor que contiene todos los archivos versionados, y varios clientes que descargan los archivos de ese lugar central. Durante muchos años, éste ha sido el estándar para el control de versiones.

Sistemas de control de versiones distribuidos

Es aquí donde entran los sistemas de control de versiones distribuidos (Distributed Version Control Systems o DVCSs en inglés). En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no sólo descargan la última instantánea de los archivos: replican completamente el repositorio. Así, si un servidor muere, y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios de los clientes puede copiarse en el servidor para restaurarlo. Cada vez que se descarga una instantánea, en realidad se hace una copia de seguridad completa de todos los datos

Fundamentos de Git

La principal diferencia entre Git y cualquier otro VCS (Subversion y compañía incluidos) es cómo Git modela sus datos. Conceptualmente, la mayoría de los demás sistemas almacenan la información como una lista de cambios en los archivos.

Git no modela ni almacena sus datos de este modo. En cambio, Git modela sus datos más como un conjunto de instantáneas de un mini sistema de archivos. Cada vez que confirmas un cambio, o guardas el estado de tu proyecto en Git, él básicamente hace una foto del aspecto de todos tus archivos en ese momento, y guarda una referencia a esa instantánea. Para ser eficiente, si los archivos no se han modificado, Git no almacena el archivo de nuevo —sólo un enlace al archivo anterior idéntico que ya tiene almacenado—.

Toda la información acerca de ***git*** que ha sido presentada cumple para fines informativos, se puede encontrar una guía completa y referencia en: http://git-scm.com/book/es, que es el sitio de distribución oficial del sistema de control de versiones.

A continuación se listan todos los comandos que son utilizados con mayor frecuencia y ayudan en el trabajo cotidiano con git.

Comando	Descripción
git init	Inicia un nuevo repositorio git
git add	Agrega un archivo o grupo de archivos al índice antes de ser enviados.
git commit –m “comentario”	Confirma los cambios y envía a git los archivos añadidos con *git add*
git push	Publica los cambios en algún servidor que almacene el proyecto git. Ejemplo: github.com
git pull	Llama todos los cambios que se encuentren en algún servidor en internet, ejemplo: github.com
git merge *RAMA*	Combina cambios hechos en alguna rama con la rama actual
git branch *NOMBRE*	Crea una ramificación o rama donde se trabajara alguna funcionalidad nueva sin afectar la rama estable
git checkout *NOMBRE*	Cambia de una rama a otra.
git remote add *NOMBRE URL*	Agrega una dirección remota en donde se encuentra almacenada una copia de nuestro trabajo.

2.2.5 Arquitectura de Zend Framework

Zend Framework (ZF) es un marco para el desarrollo de aplicaciones web utilizando el lenguaje PHP 5 es totalmente Open Source (código abierto), contiene librerías que ayudan al manejo de Base de Datos (DAO – Data Access Object), Sesiones, Autenticación, Validación, Filtros, Manejo de correos, Internacionalización, Webservice (XML RPC o REST) entre otros. La filosofía utilizada en Zend es escribe una vez y utiliza muchas veces, por lo que ellos animan a dedicarle tiempo al núcleo de nuestro desarrollo y no concentrarnos en cómo hacer cosas que ya están hechas.

Para ZF existen dos formas de utilizar, la primera consiste en usarlo como una librería que provee los componentes para desarrollar aplicaciones, la segunda consiste en utilizarlo como un marco de trabajo para el desarrollo de aplicaciones con el propósito de sacarle provecho y el potencial completo.

Por lo consiguiente se ha de trabajar con su forma de marco de trabajo, la cual está constituida por una estructura especial de directorios y archivos, que permiten generar aplicaciones Modulares y aplicando MVC.

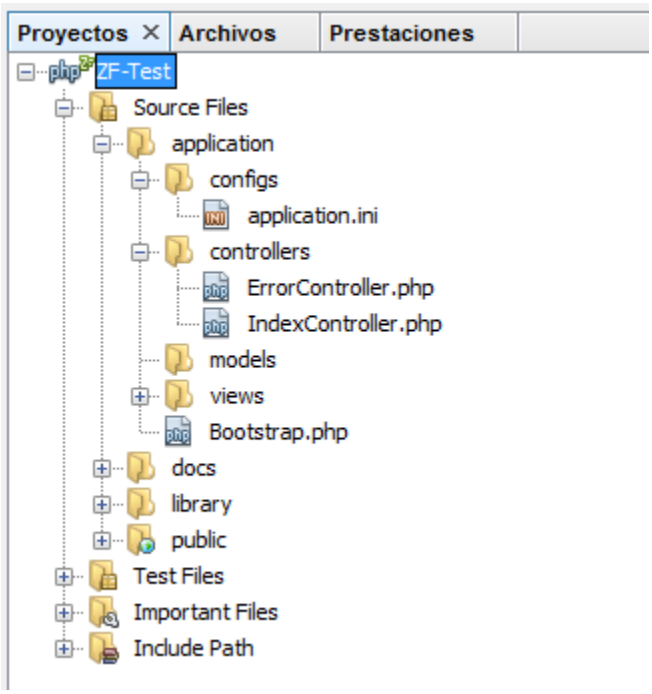


Ilustración 9 Estructura básica de una aplicación con ZF

Pero si necesitamos hacer una aplicación más potente, con capacidad de extensión, de trabajo a largo plazo hay que optar por la estructura modular de Zend Framework, la cual está dada por la siguiente estructura de directorios.

La estructura básica modular permite construir módulos que se pueden agregar con el paso del tiempo, también contribuye a separar mucho más cada segmento del sistema por lo que el trabajo se torna mucho más organizado y el desarrollo puede ser incremental.

Cada módulo es independiente del otro por lo que si un módulo falla los demás no se veran afectados.

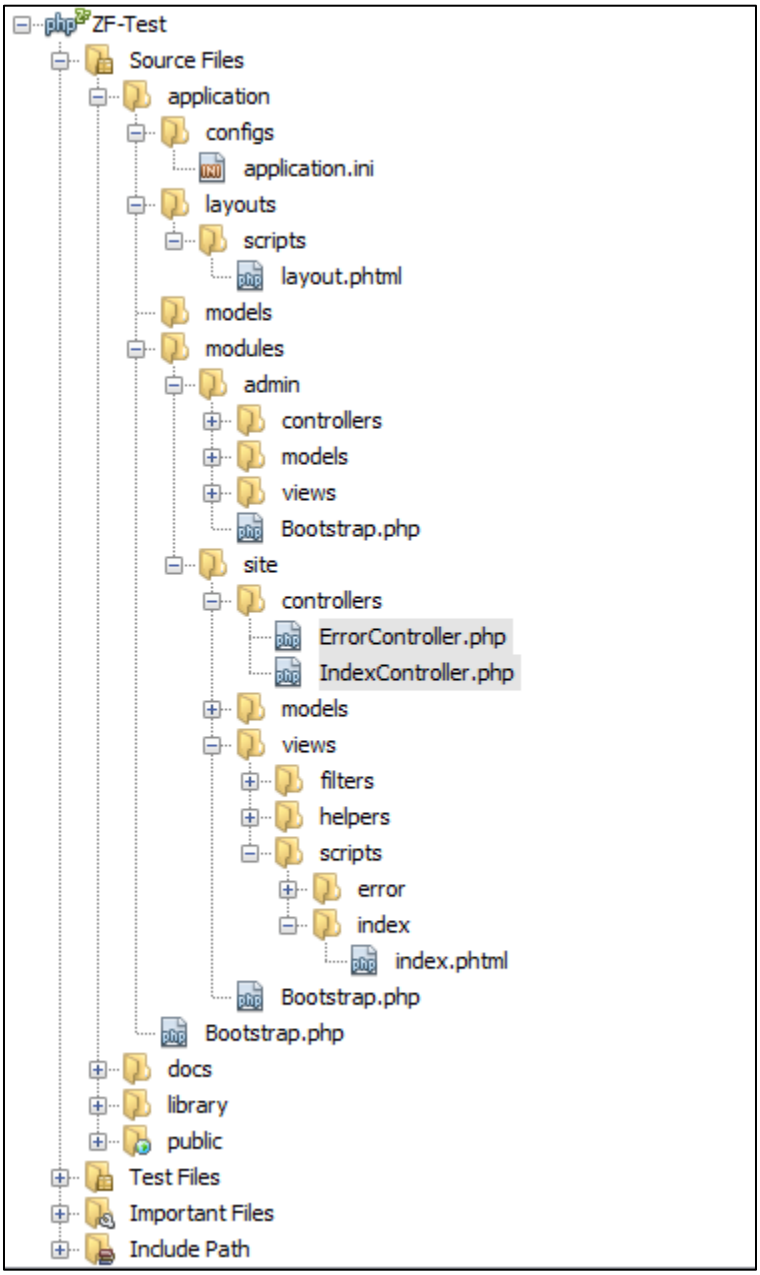


Ilustración 10 Estructura modular básica para ZF

Capítulo III Análisis

Desarrollar aplicaciones web conlleva a realizar una serie de pasos antes de comenzar a escribir el código fuente de esta.

3.1 Desarrollo de aplicaciones web sin usar un framework

Por un lado tenemos a quienes comienza a aprender a programar aplicaciones web de libros, manuales básicos y/o esbozos que escriben personas entusiastas sobre el tema; sin embargo, ellos no buscan aprender metodologías de desarrollo que les ayuden a ser más eficientes en su trabajo y se conforman con lo que han encontrado.

Un desarrollador novato normalmente para comenzar a desarrollar realiza las siguientes tareas:

- Instalar un servidor web local.
- Instalar un IDE para desarrollar (Netbeans por ejemplo).

También encontramos que desarrollan siguiendo una estructura similar a la siguiente:

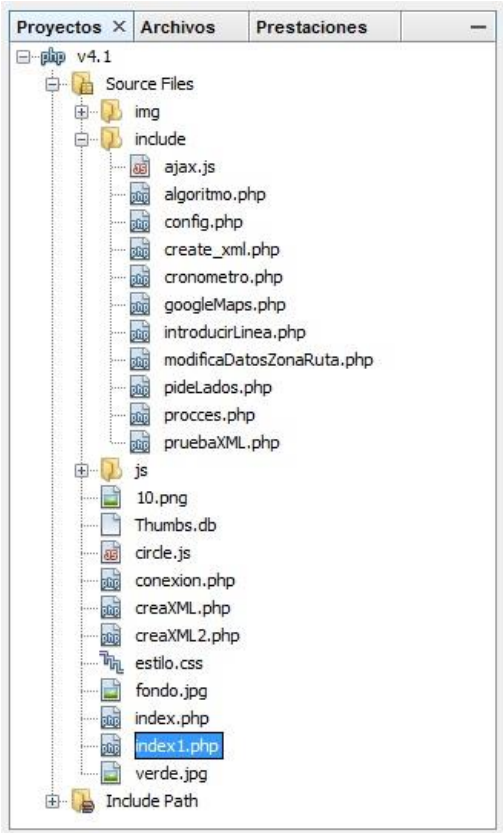


Ilustración 11 Organización típica de los archivos cuando se aprende a programar aplicaciones web

3.1.1 Ciclo de desarrollo de una aplicación web

El proceso de desarrollar una aplicación web con PHP puro lo realizan de acuerdo al siguiente esquema:

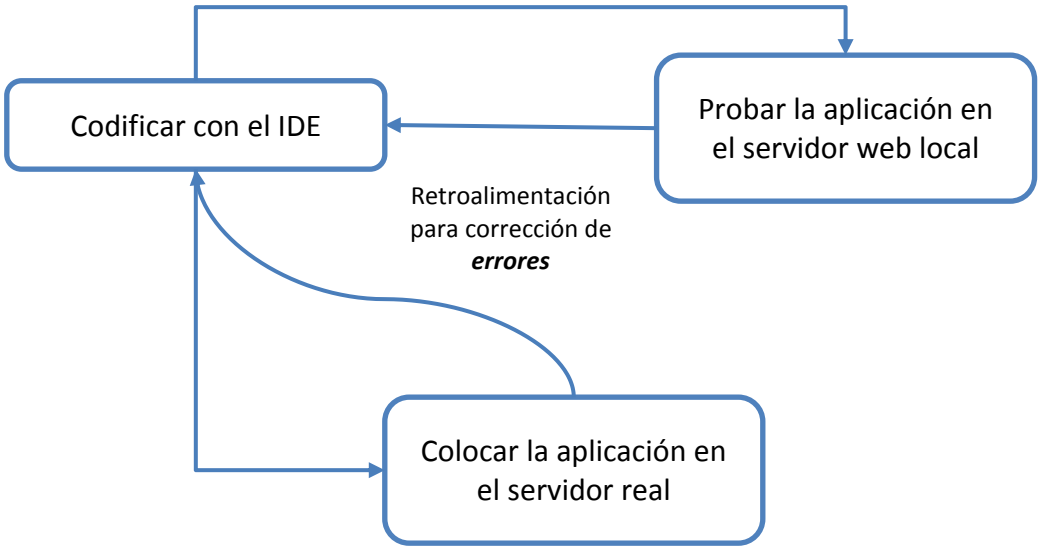


Ilustración 12 Ciclo de desarrollo

Algunos de los errores que llegan a presentarse en las aplicaciones desarrolladas sobre este modelo tienen que ver con la incompatibilidad y diferencias entre el servidor local y el servidor real en donde se coloca la aplicación.

Cabe mencionar que el tiempo de desarrollo puede ser algo largo debido a que tienen que escribir tareas o rutinas necesarias del sistema ya que normalmente todo lo comienzan desde cero.

3.2 Desarrollo de aplicaciones web usando frameworks

En contraste podemos encontrar a los desarrolladores que comienzan a aventurarse con algún Framework como apoyo para el desarrollo de su aplicación, quienes realizan siguientes tareas antes de comenzar a desarrollar:

- Seleccionar el Framework y descargarlo.
- Instalar un servidor web local.
- Instalar un IDE (Netbeans).

Existen frameworks para desarrollar en PHP y cada uno tiene su forma peculiar para la organización de sus archivos, pero comparten la utilización de MVC.

3.2.1 CakePHP

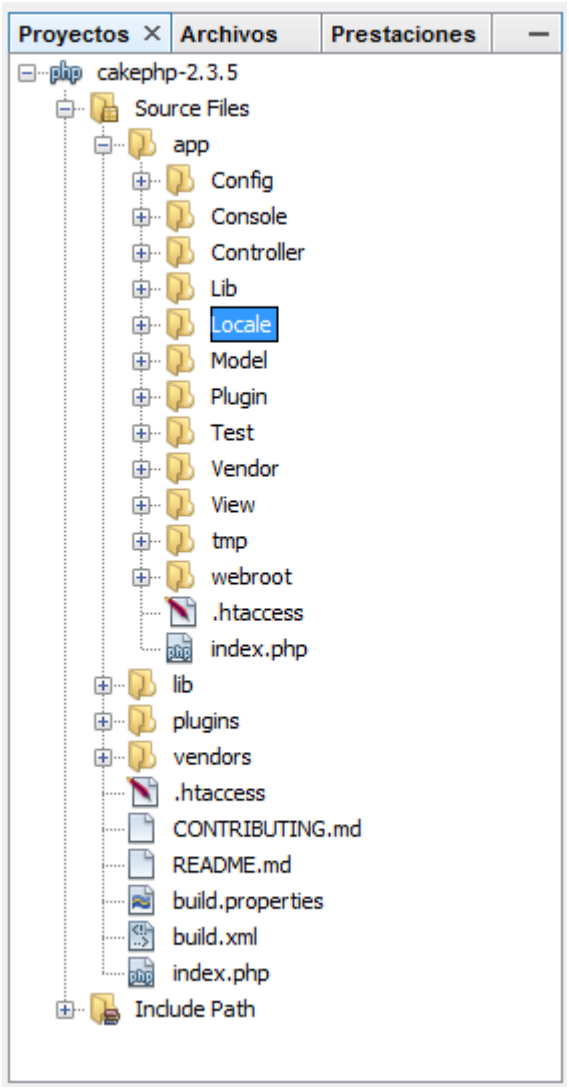
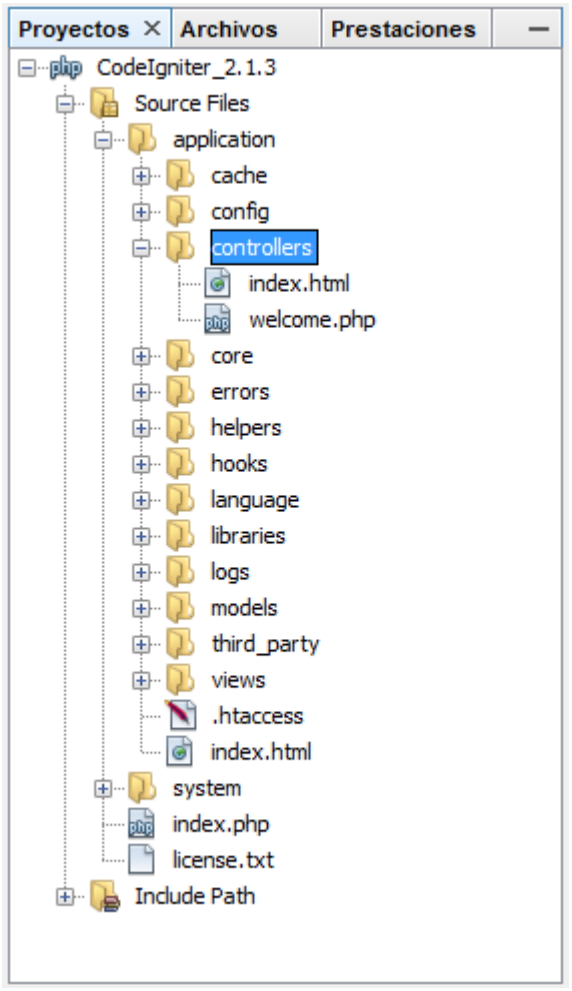


Ilustración 13 CakePHP y su organización

El framework CakePHP tiene una organización que permite identificar en donde colocar cada elemento que se ha de desarrollar. Utilizado para desarrollar aplicaciones web con complejidad media a alta.

3.2.2 CodeIgniter



CodeIgniter

Ilustración 14 CodeIgniter y su organización

Un framework más ligero que CakePHP pero no menos potente, está pensado para el desarrollo rápido de aplicaciones web que van desde complejidad baja a media.

3.2.3 Zend Framework

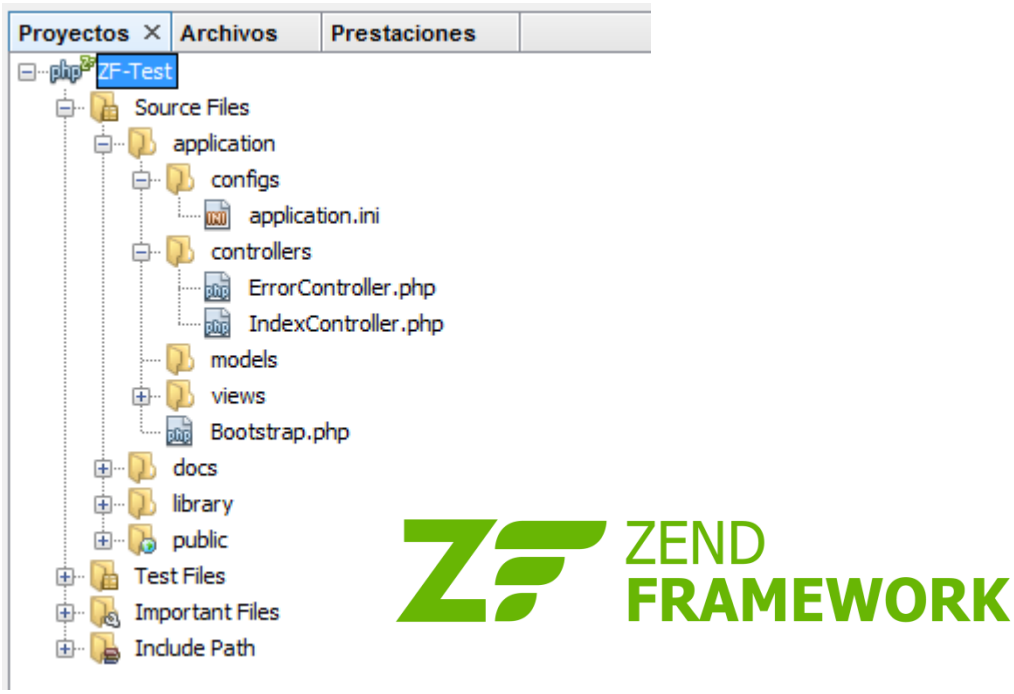


Ilustración 15 Zend Framework 1.12 y su organización

A diferencia de los anteriores frameworks, Zend Framework tiene la capacidad de trabajar de forma modular, por lo que una aplicación web puede ser desarrollada en base a módulos. Para más información ir a la sección *2.2.5 Arquitectura de Zend* Framework.

3.2.4 Ciclo de desarrollo usando un Framwork

Finalizados los pasos pueden comenzar a desarrollar, por lo que podemos encontrar el siguiente ciclo de trabajo:

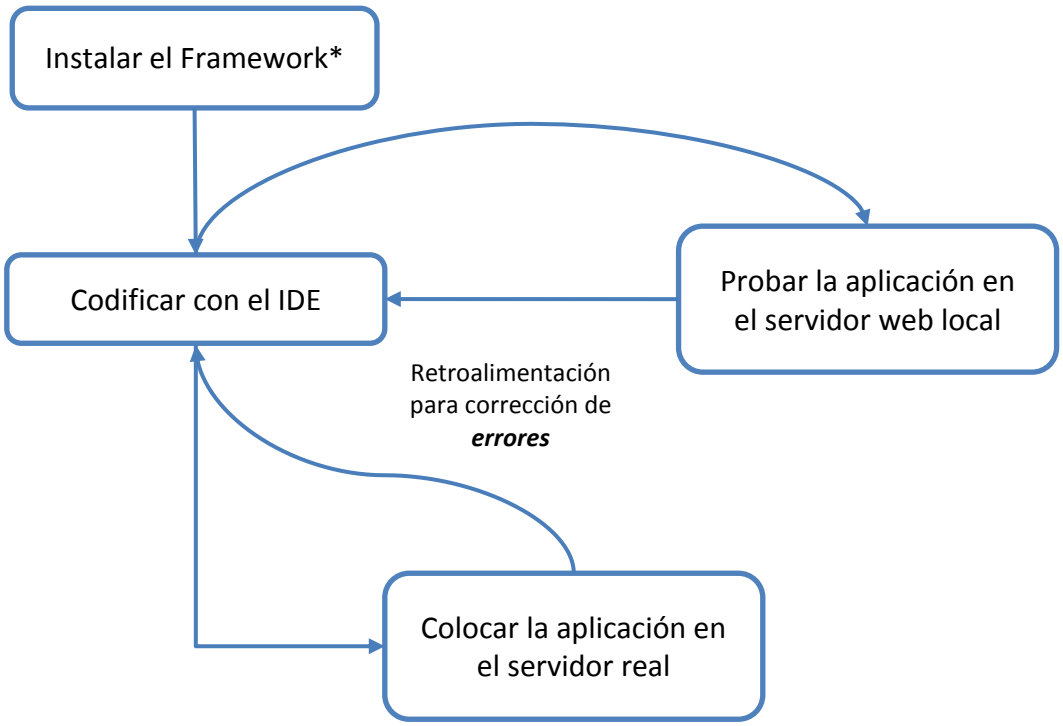


Ilustración 16 Desarrollando con un Framework

*Dependiendo del Framework que se haya escogido el método de instalación y utilización cambia.

Con el uso de un Framework se reduce el tiempo de desarrollo, puesto que en él se encuentran muchas de las rutinas más usadas para crear una aplicación web y permite concentrarse en el desarrollo de la aplicación y sus partes.

Capítulo IV Propuesta de desarrollo

Para resolver el problema de la falta de organización dentro de un proyecto y ayudar a su rápido desarrollo, es recomendable utilizar un framework y seguir su estructura recomendada para la organización de los archivos.

Existen muchos frameworks y los más usados ya han sido mencionados, cada uno propone una forma de cómo organizar los archivos que pertenecen al desarrollo. No podemos decir que forma de organizar es la mejor pero es válido aceptar las propuestas, por lo que se opta por usar Zend Framework y su estructura recomendada de directorios, y sobre todo por la facilidad para realizar el desarrollo en base a módulos.

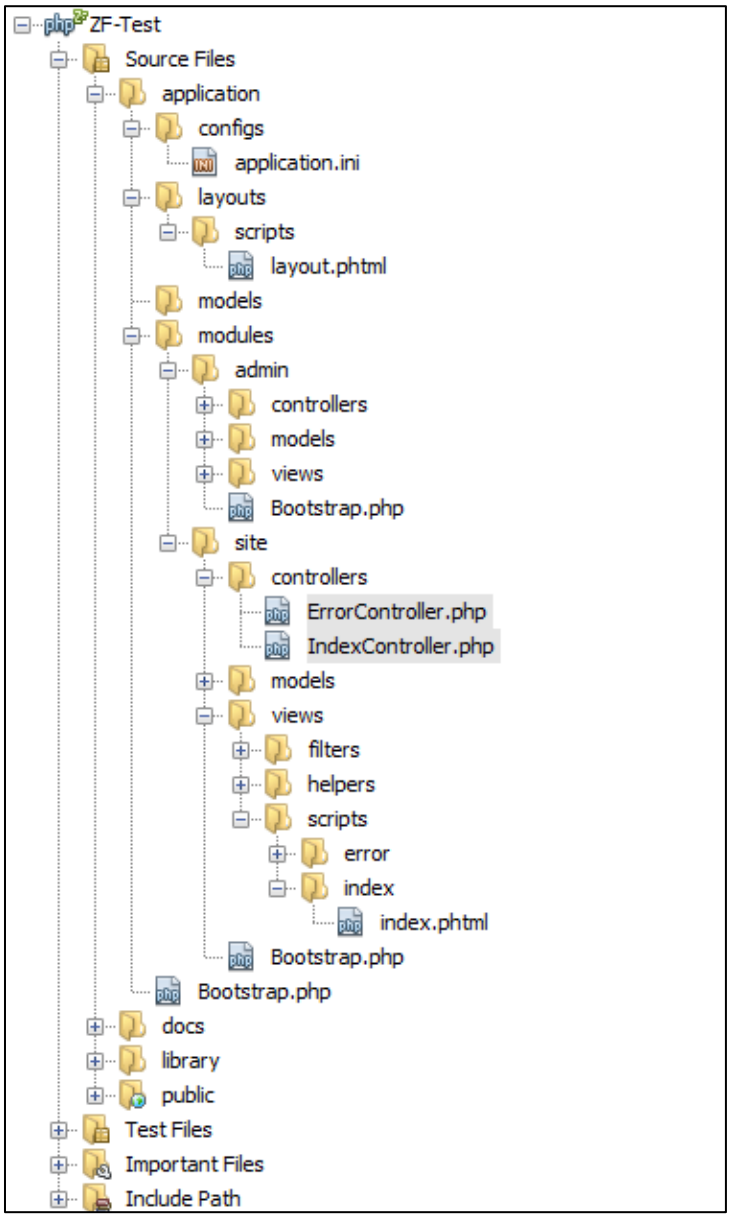


Ilustración 17 Estructura modular

Para poder resolver la pérdida de código fuente y llevar un control de productividad en base a saber quién y con qué frecuencia colaboran en el proyecto, se recomienda usar un sistema de control de versiones ya que estos tienen la capacidad de anotar quien ha hecho las modificaciones, esto se puede completar usando un servidor de control de versiones como GitHub o Bitbucket y así realizar un trabajo en equipo.

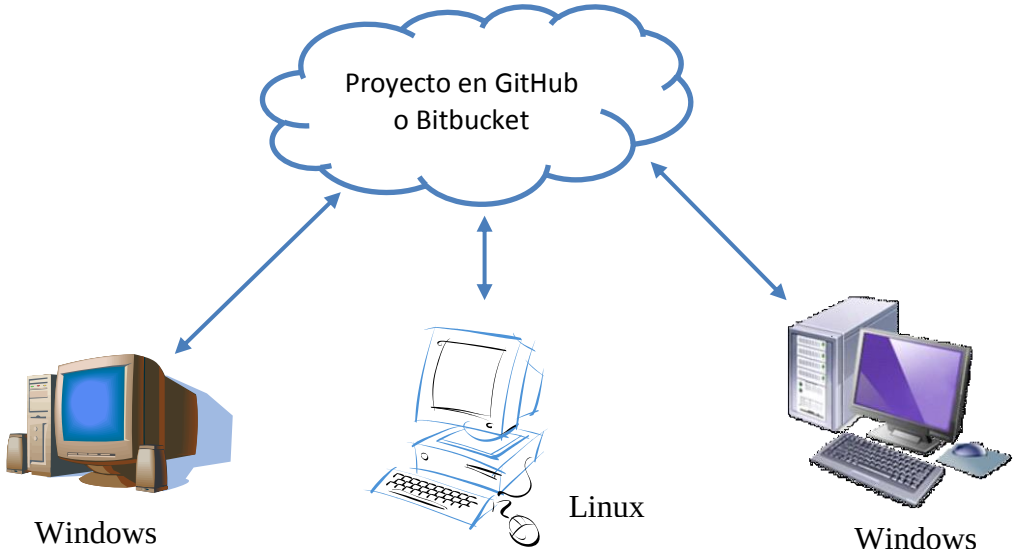


Ilustración 18 Diagrama de trabajo en equipo

Al almacenar nuestro proyecto en Github o Bitbucket

Almacenamiento del código fuente permite conocer y recuperar código que se ha modificado con el paso del tiempo.

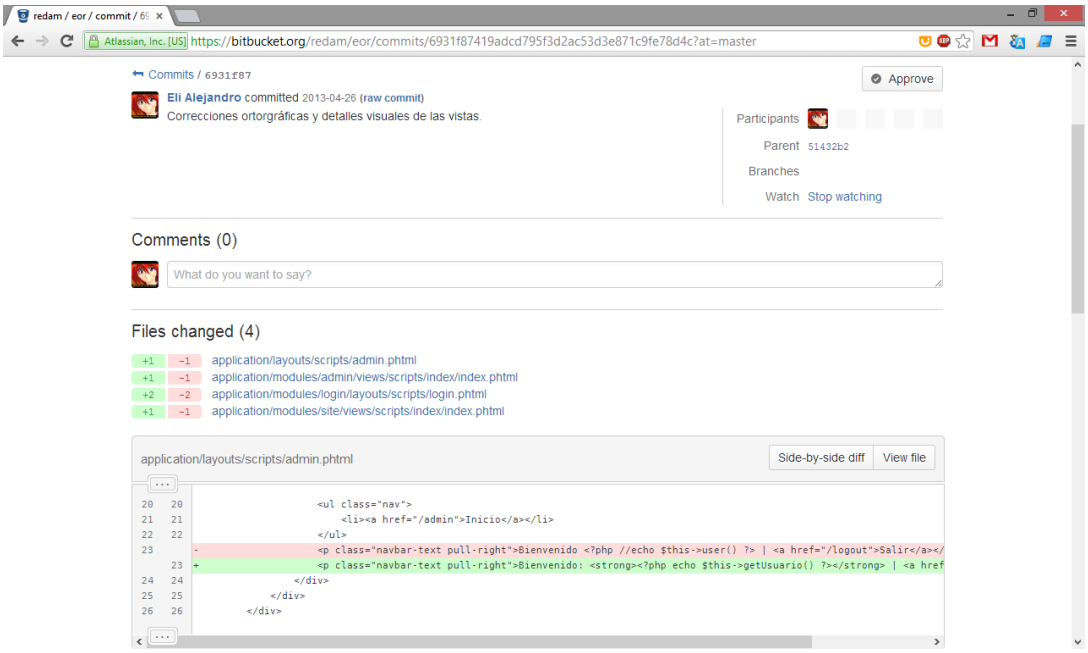


Ilustración 19 Modificaciones del código

También es posible conocer quien ha realizado modificaciones, que archivo y parte se ha modificado.

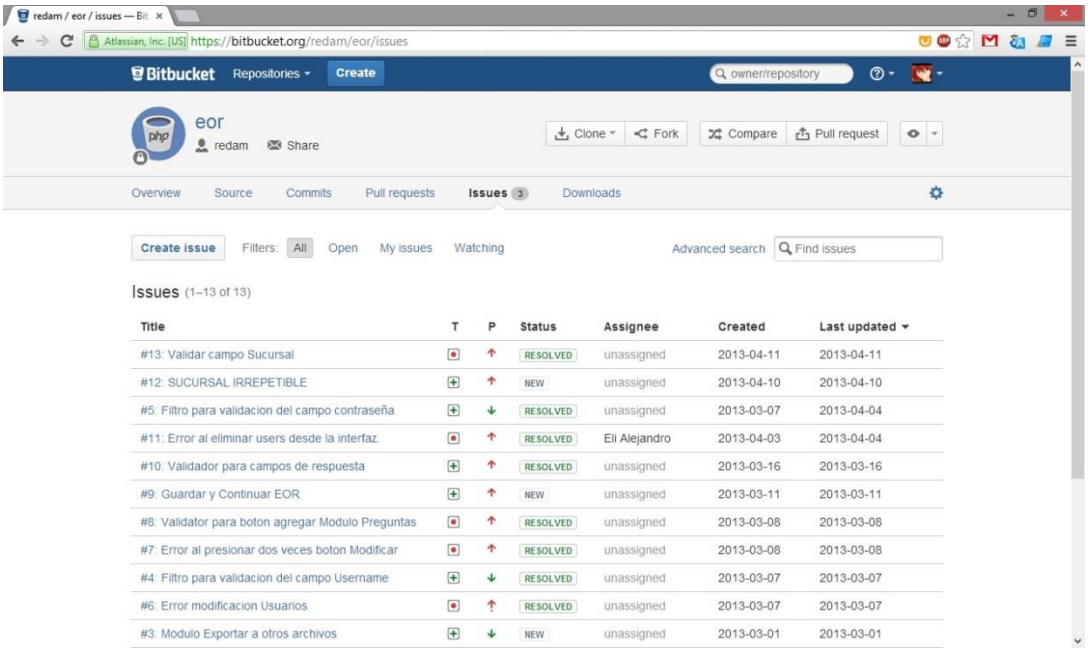


Ilustración 20 Seguimiento del Desarrollo

Con el control de seguimiento se puede llevar nota de las metas planeadas para el sistema así como el control del reporte de errores y correcciones.

Trabajar en equipo implica que el proyecto puede estarse desarrollando sobre sistemas operativos diferentes (Windows, Linux o Mac OS X), y esto significa que cada sistema debe de configurarse conr un servidor web para poder ejecutar el código PHP; para esta tarea se cuenta con paquetes preconfigurados como: WampServer (Windows), xampp (distribuciones a medida para cada sistema), lamp (instalación de componentes individuales en Linux), MAMP (Mac OS X); al usar paquetes preconfigurados, se asume que algunos componentes funcionaran correctamente y otros simplemente no lo harán porque no son compatibles con el sistema operativos.

Para poder lograr un mayor grado de compatibilidad y homogenizar el entorno de pruebas local, es necesario que todo el equipo cuente con una plataforma de pruebas similar tanto para el equipo como que se asemeje al entorno de producción real.

Para solventar este problema se recomienda usar una plataforma como lo es Vagrant, la cual permite generar máquinas virtuales para ejecutar el código PHP y estas son las mismas que utilizará el equipo de desarrollo.

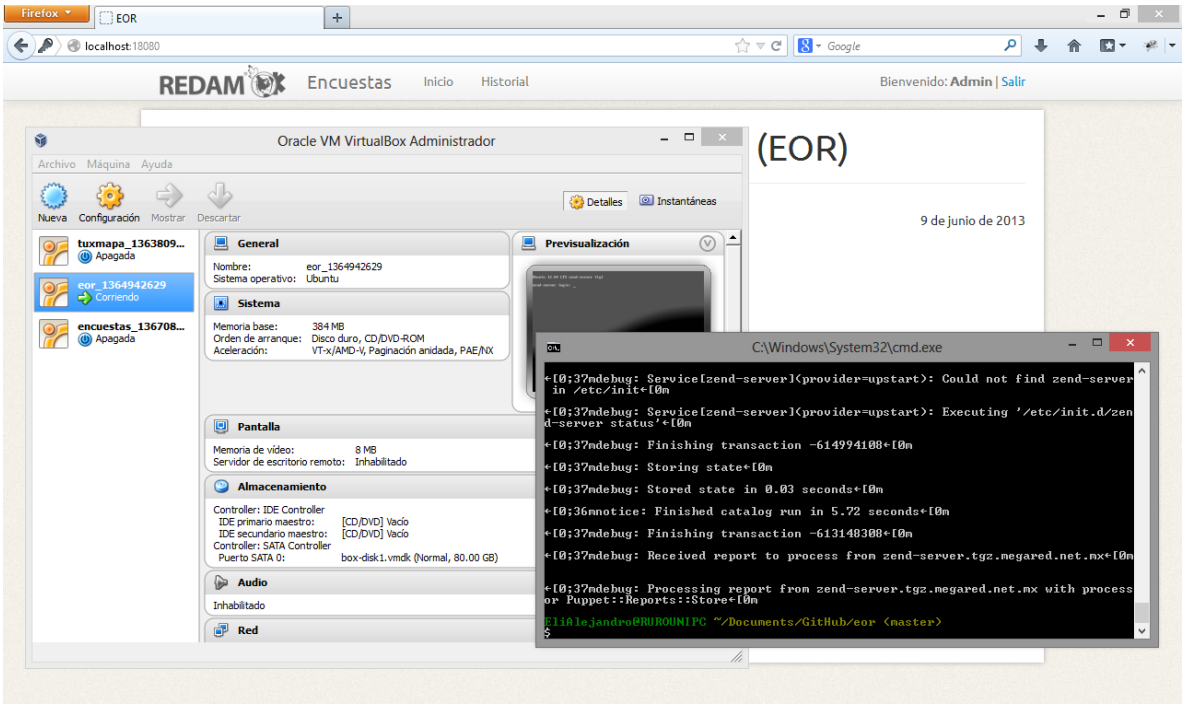


Ilustración 21 Equipo virtual en ejecución, Vagrant configurado con Zend Server

Capítulo V Desarrollo

Sistema de Encuestas Online de Redam (EOR)

Es una aplicación web que está enfocada a la captura de encuestas de evaluación para las actividades realizadas dentro de cada sucursal de la empresa, entre estas actividades se encuentran: limpieza y atención al cliente. Surge debido a la necesidad de evitar la duplicación de información, mejorar los tiempos en que se tienen los gráficos y facilitar el acceso a los informes generados.

Para llegar al estado actual del sistema se ha llevado alrededor de tres meses, comprende el análisis del problema, diseño de la base de datos y codificación hasta el estado actual.

Ilustración 22 Pantalla de Inicio de EOR

EOR es un sistema que va creciendo conforme se han completado las tareas principales, esto que quiere decir, que no es necesario que sea el sistema final para que pueda comenzar a utilizarse y generar los primeros reportes.

Para el desarrollo de EOR se ha utilizado Zend Framework y su arquitectura de módulos para permitir el desarrollo aislado de cada componente y/o sección del sistema, así como su desarrollo incremental.

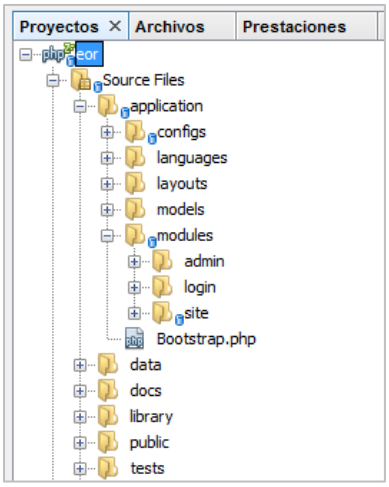


Ilustración 23 Arquitectura de directorios para EOR

Para el desarrollo del sistema se ha utilizado Vagrant y Git, por lo que es posible continuar su desarrollo desde cualquier sistema operativo sin la necesidad de configurar un servidor web local, basta con tener instalado git, ruby, VirtualBox y Vagrant, los cuales no son difíciles de instalar salvo hacer una pequeña configuración a git.

En git descargar un repositorio se le llama clonar, es decir, descargar una copia a nuestro equipo.

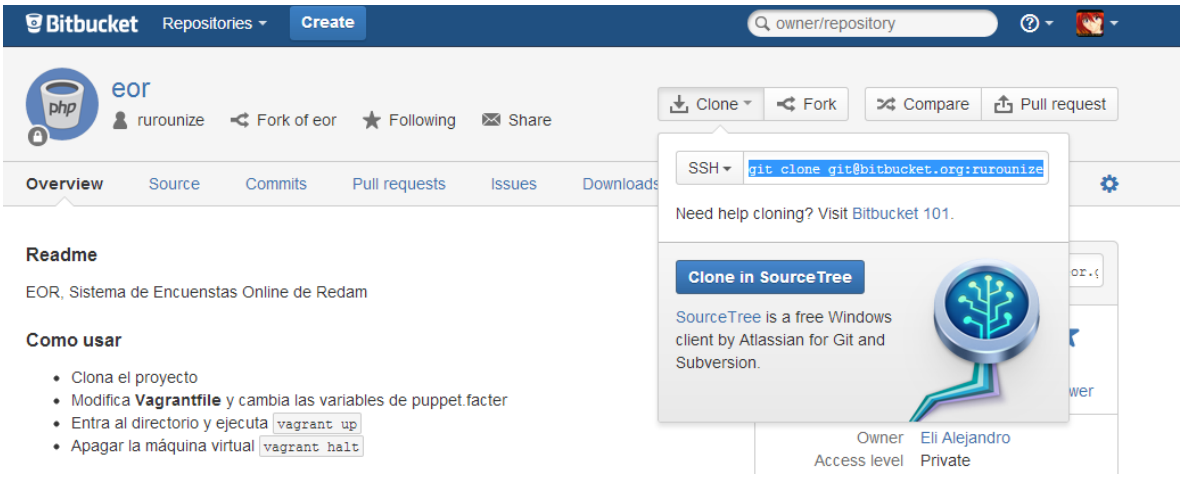


Ilustración 24 Repositorio y dirección para clonar el repositorio

Clonando una copia del sistema a nuestro equipo local

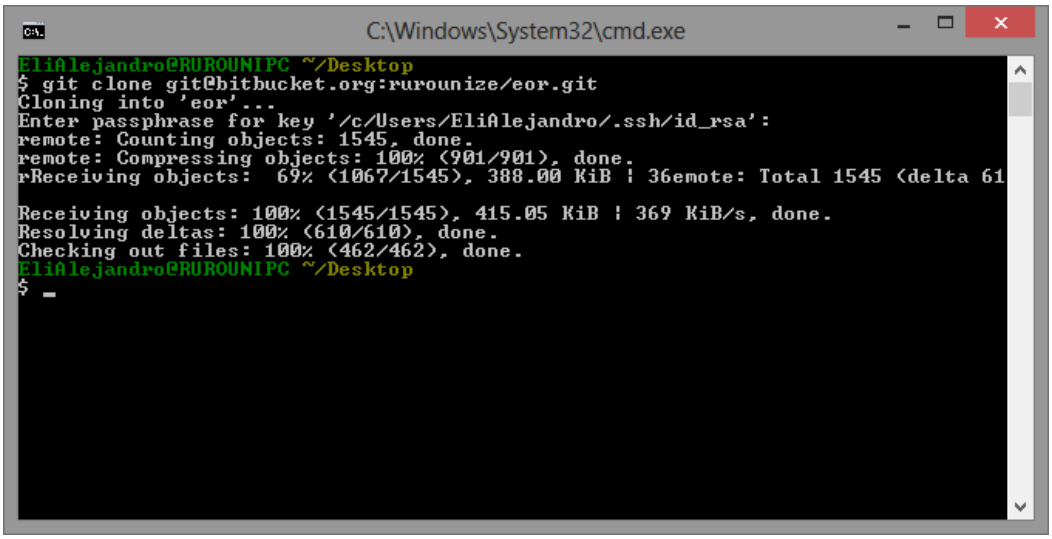


Ilustración 25 Clonando una copia a nuestro equipo local

Para continuar el desarrollo es necesario iniciar *vagrant* por lo que se utiliza el comando *vagrant up*

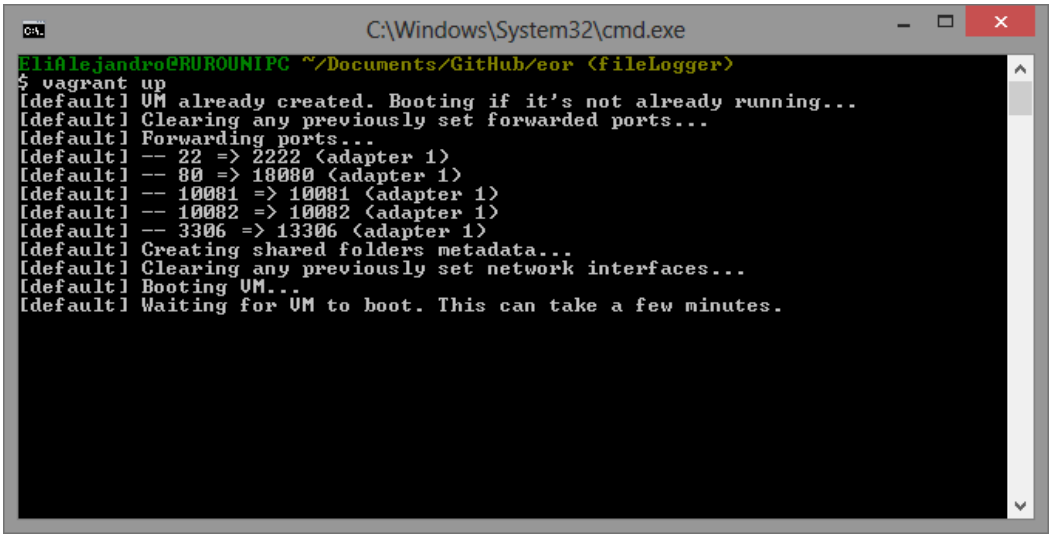


Ilustración 26 Iniciando Vagrant

El modelo de distribución del código fuente para su desarrollo es muy práctico y garantiza que las características de los equipos de prueba sean similares, incluso comparables con las que se encontraran en el servidor de producción. También se garantiza que no se altere un trabajo estable al crear nuevas características que pueden ser inestables, esto se logra creando una nueva *branch* (rama) con git.

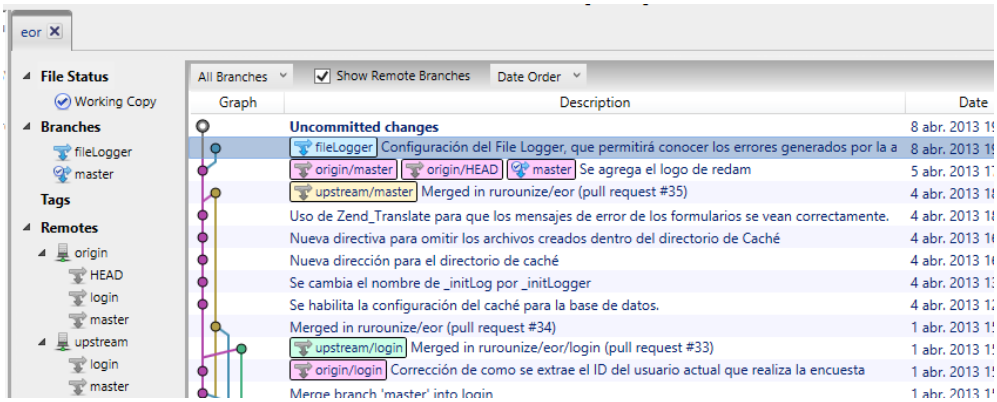


Ilustración 27 Modo gráfico de git: Ramas dentro del desarrollo de EOR.

Cuando se trabaja con ramas de desarrollo, al combinar los cambios estables en la rama principal (master) se pueden ver los gráficos del desarrollo de la siguiente manera.

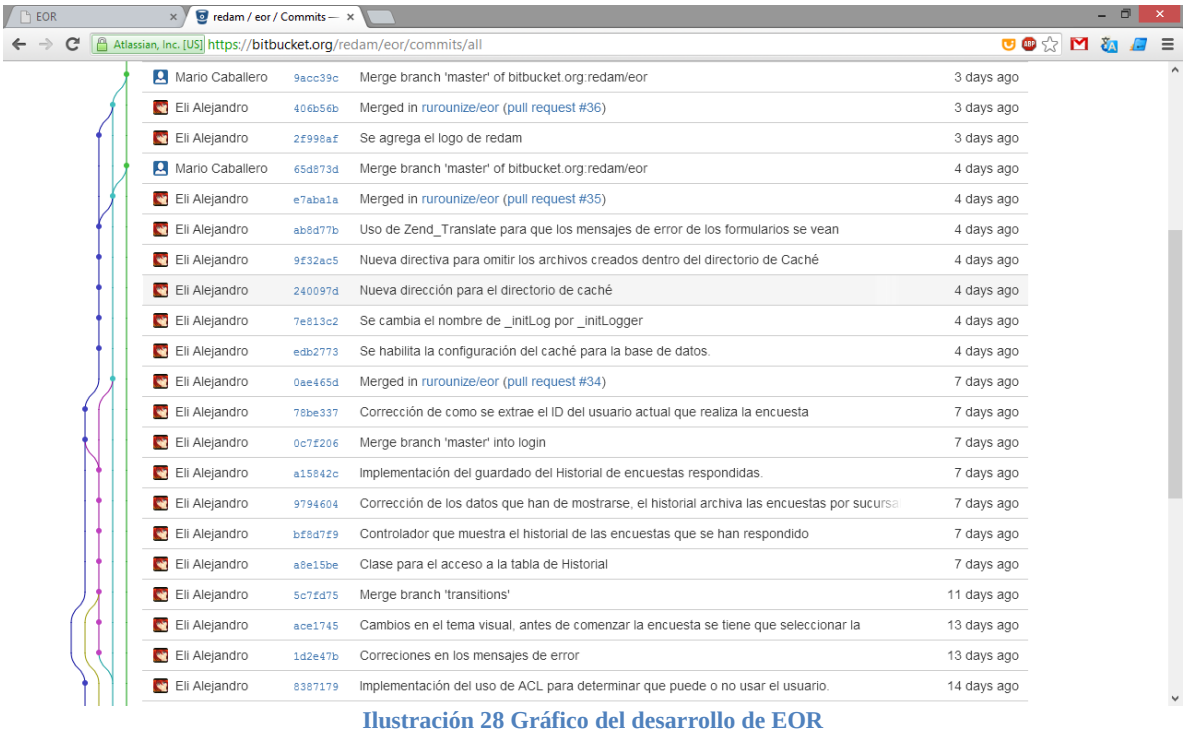


Ilustración 28 Gráfico del desarrollo de EOR

Conclusiones. (Resultados)

Analizando cada aplicación presentada se llega a los resultados, en estos se ven reflejados *pros* y los *contras* de la utilización de cada estilo de programación, usar patrones de diseño o no, usar un framework o no usarlo.

En la siguiente tabla se comparan algunos aspectos importantes al desarrollar en PHP puro y desarrollar usando un marco de trabajo como *Zend Framework.*

	php puro	**Zend Framework**
Facilidad de aprendizaje	✓	×
Tiempo de desarrollo	×	✓
Separación de código HTML y PHP (MVC)	×	✓
Facilidad de cambiar entre sistemas de bases de datos (MySQL, Postgre, SQL Server)	×	✓
Sentencias SQL en cada página usada	✓	×
Programación orientada a Objetos	✓	✓
Facilidad de hacer cambios en sentencias SQL	✓	✓
Facilidad de hacer cambios en sentencias SQL de múltiples páginas	×	✓
Uso de patrones de diseño	×	✓
Reutilización de código	×	✓

También podemos comparar la forma en que se construye cada página según el estilo de programación.

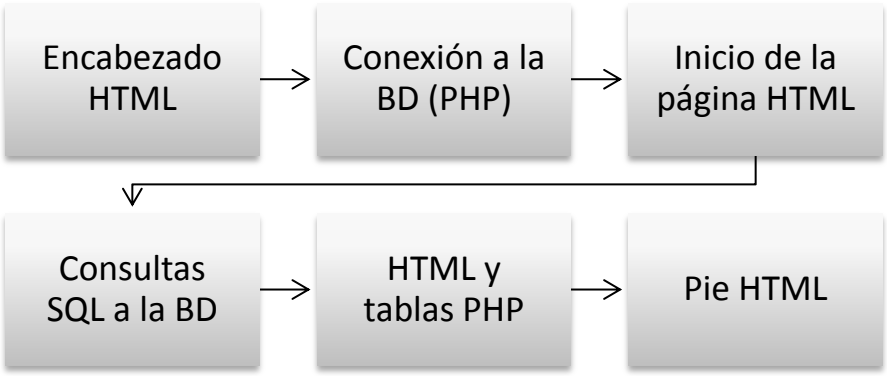


Ilustración 29 Organización típica de un archivo PHP creado por novatos, intercarla código HTML y PHP

Por otro lado, encontramos el diagrama de flujo de cómo se organiza una aplicación creada con Zend Framework.

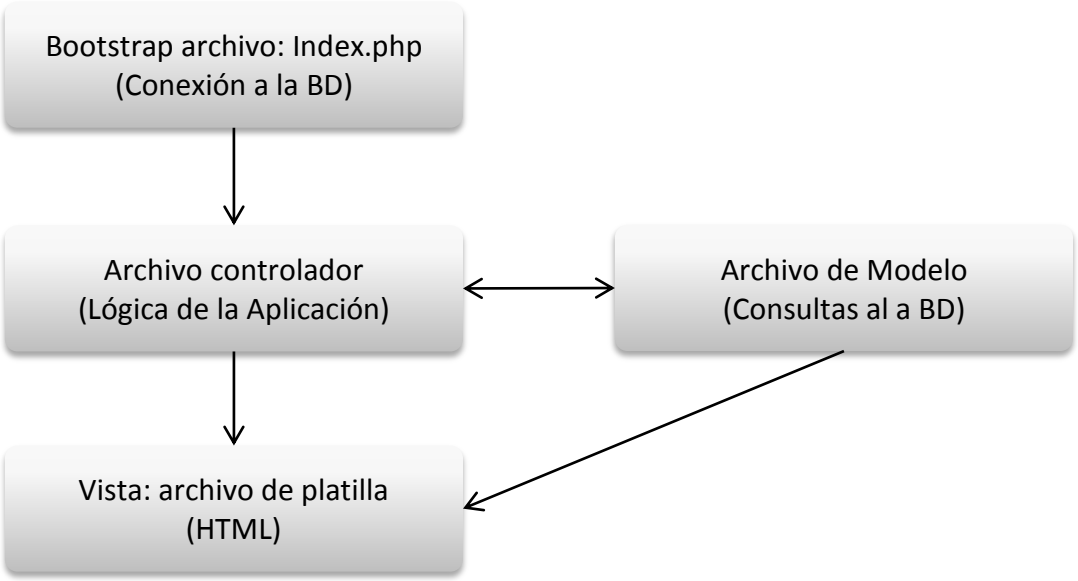


Ilustración 30 Organización de una aplicación típica usando MVC con Zend Framework

Gracias a la aplicación de una Arquitectura como la de Zend Framework se logró una mejor organización del proyecto y simplifica su modificación, con la creación de aplicaciones modulares se logra mayor independencia dentro del sistema y se permite el crecimiento progresivo del mismo.

Ya que Zend Framework está pensado para el desarrollo rápido de aplicaciones (Rapid Application Development o RAD) se logra una mejora en el tiempo de desarrollo de aplicaciones web a diferencia de escribir aplicaciones con PHP puro.

Con la utilización de *Git* que permite llevar un control de versiones es posible hacer modificaciones no destructivas dentro del sistema, además permite regresar a versiones anteriores por lo que siempre se tendrá un respaldo del código fuente original.

Bibliografía

Alexander, C. (1977). *A Pattern Language. An inspiration for many people in the patterns movement. I'm less enamored of it than most, but it's worth.* Oxford: Oxford University Press.

Atlassian. (2013). *Bitbucket Documentation*. Retrieved Mayo 30, 2013, from Bitbucket Documentation: https://confluence.atlassian.com/display/BITBUCKET/Bitbucket+Documentation+Home;jsessionid=3EBE1F147FF0B1093C62180DD6AC20C5

Bourdon, R. (2013). *WampServer*. Retrieved Mayo 31, 2013, from WampServer: http://www.wampserver.com/en/

Cake Software Foundation, Inc. . (2013). *CakePHP*. Retrieved Mayo 23, 2013, from CakePHP: http://book.cakephp.org/2.0/es/index.html

Chacon, S. (2009). *Pro GIT* (Primera ed.). United States of America: Apress.

Collins-Sussman, B., Fitzpatrick, B. W., & Pilato, C. (2004). *Version Control with Subversion.* O'Reilly Media.

CVS HOME. (n.d.). *CVS HOME*. Retrieved Abril 19, 2013, from CVS HOME: http://www.cvshome.org/eng/

EllisLab. (2013). *Codeigniter / Ellislab*. Retrieved Mayo 2013, 23, from Codeigniter / Ellislab: http://ellislab.com/codeigniter

Elmasri, R., & Navathe, S. B. (1997). *Sistemas de Bases de Datos.* Wilminton, Delaware; EUA: Addison-Wesly Iberoamericana, S.A.

Engine Yard. (2013). *About Engine Yard, Inc.* Retrieved Enero 14, 2013, from Engine Yard: https://www.engineyard.com/company

JavaWorld. (2000). *HMVC: The layered pattern for developing strong client tiers*. Retrieved Noviembre 12, 2012, from Javaworld: http://www.javaworld.com/javaworld/jw-07-2000/jw-0721-hmvc.html

MAMP. (2013). *MAMP: Mac - Apache - MySQL*. Retrieved Mayo 31, 2013, from MAMP: Mac - Apache - MySQL: http://www.mamp.info/en/mamp/index.html

Microsoft MSDN. (2013). *Modularity*. Retrieved Abril 15, 2013, from MSDN Modularity: http://msdn.microsoft.com/en-us/library/ff921069(v=pandp.20).aspx

MySQL. (2011). *MySQL :: MySQL 5.0 Reference Manual*. Retrieved Noviembre 8, 2012, from MySQL Dev: http://dev.mysql.com/doc/refman/5.0/es/what-is.html

Oracle Corporation and/or its affiliates. (2012). *Netbeans.org*. Retrieved Noviembre 08, 2012, from Netbeans.org: http://netbeans.org/index_es.html

Ruby Lang Org. (2013). *Lenguaje de Programación Ruby*. Retrieved Marzo 8, 2013, from Lenguaje de Programación Ruby: http://www.ruby-lang.org/es

The PHP Documentation Group. (2012). *Prefacio*. Recuperado el 08 de Noviembre de 2012, de PHP: http://www.php.net/manual/es/preface.php

W3 org. (2002, Marzo 4). *W3/Protocols*. Retrieved Abril 15, 2013, from W3/Protocols: http://www.w3.org/Protocols/Specs.html

Wikipedia. (2012). *Framework - Wikipedia*. Retrieved Noviembre 8, 2012, from Wikipedia: http://es.wikipedia.org/wiki/Framework

Wikipedia. (2012). *Open Source (Código Abierto)*. Recuperado el 08 de Noviembre de 2012, de Wikipedia: http://es.wikipedia.org/wiki/C%C3%B3digo_abierto

Wikipedia. (2012). *Servidor HTTP Apache - Wikipedia*. Retrieved Noviembre 8, 2012, from Wikipedia: http://es.wikipedia.org/wiki/Servidor_HTTP_Apache

Zend. (2012). *Zend Framework & MVC Introducction - Zend Framework Quick Start*. Retrieved Noviembre 9, 2012, from Zend Framework: http://framework.zend.com/manual/1.12/en/learning.quickstart.intro.html#learning.quickstart.intro.mvc

Zend Framework. (2012). *Zend Framework*. Retrieved Noviembre 08, 2012, from Zend Framework: http://framework.zend.com/manual/1.12/en/introduction.overview.html