



Espíritu Emprendedor TES 2018, Vol 2, No. 3, julio a septiembre 34-53
Artículo De Revisión Bibliográfica

DOI: <https://doi.org/10.33970/eetes.v2.n3.2018.93>

Indexada Latindex ISSN 2602-8093

<https://www.espirituemprededortes.com/>

Comparación de usabilidad y complejidad de frameworks: symfony, cakephp y zend framework

Ing. Marcos Antonio Espinoza Mina, Mgs.

Universidad Ecotec / Universidad Agraria del Ecuador, Ecuador

Autor para la correspondencia: mespinoza@ecotec.edu.ec, mespinoza@uagraria.edu.ec

Jorge Peralta Godoy

Universidad Tecnológica Ecotec, Ecuador

Autor para la correspondencia: jorgedpg10@gmail.com

Fecha de recepción: 2 de mayo de 2018 / **Fecha de aceptación:** 23 de julio de 2018

Resumen.

Actualmente, existen cientos de frameworks para producir contenido web en PHP (Hypertext Pre-Processor), tres se destacan, estos son: Symfony, CakePHP y Zend Framework; los cuales tienen la ventaja de tener licencias libres. En el presente trabajo se introdujo el concepto de framework, se estudió si es recomendable usarlo o no, sus ventajas y desventajas. El objetivo principal fue hacer un análisis de comparación documental sobre los frameworks antes mencionados, como opción para un desarrollo de calidad usando el lenguaje PHP. La investigación corresponde a un estudio bibliográfico de libros y múltiples artículos, que fueron analizados para cumplir con los objetivos del tema, y se constató las funcionalidades básicas de las tres herramientas con una aplicación elemental. Se determinó que ninguno de los frameworks supera en todas sus características a los demás, quedó demostrado que cada uno de ellos tiene fortalezas en múltiples áreas y queda a elección del usuario la que más le convenga.

Palabras claves: Framework, PHP, Symfony, CakePHP, Zend, aplicaciones web

Abstract

Currently, there are hundreds of frameworks to produce web content in PHP (Hypertext Pre-Processor), three are highlighted, these are: Symfony, CakePHP and Zend Framework; which have the advantage of having free licenses. In the present work the concept of framework was introduced, it was studied if it is advisable to use it or not, its advantages and disadvantages. The main objective was to make a documentary comparison analysis on the aforementioned frameworks, as an option for quality development using the PHP language. The research corresponds to a bibliographic study of books and multiple articles, which were analyzed to meet the objectives of the topic, and the basic functionalities of the three tools with an elementary application were verified. It was determined that none of the frameworks

Esta obra se comparte bajo la licencia Creative Common Atribución-No Comercial 4.0

International (CC BY-NC 4.0)

Revista Trimestral del Instituto Superior Tecnológico Espíritu Santo

exceeds all other features, it was shown that each of them has strengths in multiple areas and is left to the user's choice.

Key words: Framework, PHP, Symfony, CakePHP, Zend, web applications

Introducción

Aunque se puede comprender los conceptos básicos del PHP, podría estar perdiendo un componente vital del desarrollo con este lenguaje de programación, que es el uso de los frameworks. Los frameworks pueden ayudar a los equipos de desarrollo a concentrarse en requisitos específicos sin perder un valioso tiempo y energía en el diseño e implementación en funciones comunes de bajo nivel requeridas por una aplicación. Con esta herramienta, un equipo puede desarrollar proyectos mucho más rápido que si estuvieran escribiendo todo el código desde cero (Earley, 2009). Algo a considerar es que el mercado actual proporciona docenas y docenas de frameworks para elegir; pero el gran volumen de frameworks disponibles puede ser un problema (Denoncourt, 2012).

PHP ha sido un lenguaje de programación web desde principios de la década de 1990 y es un lenguaje de scripting en el servidor que todavía es ampliamente utilizado por algunas de las aplicaciones basadas en web más populares, como Facebook (Schutt & Balci, 2016).

A principios de los 90 se empezó a desarrollar aplicaciones de internet por medio de PHP, su popularidad creció rápidamente, pero inicialmente se pensó que era un lenguaje desordenado, poco formal y con muy poco aporte a la mantenibilidad, por lo que se creía que solo servía para pequeños proyectos, sin mayores mejoras y actualizaciones. En ese tiempo los desarrolladores estaban fascinados con frameworks de otros lenguajes como Spring (Java), Django (Python) y Ruby on rails (Ruby) (Porebski, 2011). Las cosas empezaron a cambiar en el año 2004, con el lanzamiento de PHP5 su soporte orientado a objetos era elegante, y era excelente para producir código de forma rápida y contar con una buena mantenibilidad (The PHP Group, 2017).

En el año 2007 era evidente que PHP no contaba solo con uno, sino con tres grandes frameworks: Symfony, CakePHP y Zend Framework, que tenían todo lo necesario para producir proyectos con las más altas exigencias. Al ser herramientas de muy buena calidad, también se convirtieron en muy populares. La elección de estas tres plataformas está fundamentada en una investigación de Porebski (2011), ver figura 1; en la que se demuestra que los tres frameworks antes mencionados son los más populares de 2006 a 2010.



Figura 1. Estudio de popularidad de frameworks de 2006 a 2010 (Porebski, 2011)

Grandes empresas de la escala de BMW, Hyundai (Cake Software Foundation, Inc., 2017), la BBC y Cisco (Rogue Wave Company, 2017) empezaron a usar los frameworks, lo que ayudó a incrementar su popularidad, y esto produjo que PHP vuelva a ubicarse como uno de los lenguajes de propósito general más utilizados de la actualidad, ver figura 2 (IEEE Spectrum, 2017).

Language Rank	Types	Spectrum Ranking
1. Python	🌐 🖥️	100.0
2. C	📱 🖥️ 🧪	99.7
3. Java	🌐 📱 🖥️	99.5
4. C++	📱 🖥️ 🧪	97.1
5. C#	🌐 📱 🖥️	87.7
6. R	🖥️	87.7
7. JavaScript	🌐 📱	85.6
8. PHP	🌐	81.2
9. Go	🌐 🖥️	75.1
10. Swift	📱 🖥️	73.7

Figura 2. Ranking de popularidad de lenguajes de programación (IEEE Spectrum, 2017)

En el ámbito educativo, los estudiantes sin exposición a los patrones de diseño y frameworks estarán mal preparados como fuerza de trabajo. Independientemente del framework elegido, existen muchos beneficios al incorporar marcos de desarrollo de aplicaciones web en el aula. Al introducir a los estudiantes a los frameworks durante su educación universitaria, obtienen no solo una apreciación de los conceptos de

diseño como separar la interfaz de usuario de la lógica y los datos subyacentes, sino también la exposición a herramientas comerciales de software que mejoran su comerciabilidad futura (Chao, Parker, & Davey, 2013).

Pero no todo lo que brilla es oro, por ejemplo en el tema de seguridad, proyectos estudiados por Colomina, Arnedo-Moreno, & Clariso (2013), en el que se incluyó a Symfony ilustran que incluso los más populares proyectos de código abierto carecen de algunas garantías simples que los protejan contra la introducción de malware. A continuación se hace un reconocimiento sobre el alcance funcional de los frameworks, y se presenta una evaluación de los tres frameworks ya referidos.

Desarrollo

Un framework de aplicaciones web es un grupo de código organizado con una arquitectura determinada que puede ser utilizada para desarrollo rápido de aplicaciones (Porebski, 2011). Por ejemplo, se puede generar una interfaz completa Create Create Update Delete (CRUD) para una tabla con 10 líneas de código CakePHP. La tabla se lee dinámicamente desde el servidor de la base de datos y se pueden realizar cambios repetidos en la tabla SQL subyacente sin tener que volver a codificar la aplicación (Biermann, 2014).

Un desarrollador no necesita construir la función básica de una aplicación, porque el framework ya la ha creado (Udomkarn & Wakahara, 2011). Los frameworks son aplicaciones parcialmente desarrolladas que se pueden acoplar a las necesidades de desarrollo propias. Su desventaja radica en que, ya que han sido mayoritariamente desarrollada por terceros, tiene muchos componentes que no pueden ser modificables fácilmente, así que pierden flexibilidad.

La ventaja principal que proporcionan los frameworks es la disminución de errores durante el desarrollo. Los frameworks se fundamentan en el principio de “*no te repitas*” (DRY, *don't repeat yourself*) que es un concepto que se resume en no “reinventar la rueda”. Dicho principio facilita el trabajo en el mantenimiento, mejoramiento y simplicidad del código. Una muestra de lo expuesto es que en Symfony el proceso de depuración de aplicaciones está muy simplificado. Cada error y notificación se informa de que todas las operaciones de la base de datos se registran e incluso el tiempo de ejecución se mide por el marco, por lo que el desarrollador puede obtener cualquier información necesaria sobre la página en segundos (Jarmolowicz, Zabierowski, & Napieralski, 2008). Otra característica importante es la apariencia limpia que aportan a los URLs (ver tabla 1).

Tabla 1. Diferencia entre las direcciones URL.

URL tradicional	URL por frameworks
<code>/animals.php?species=cats&breed=mainecoon</code>	<code>/animals/cats/mainecoon</code>

La apariencia del URL es importante pues produce una mejora tanto visual como en los resultados de los motores de búsqueda. Existen dos paradigmas para hacer aplicaciones web, por un lado la programación por frameworks y por otro lado la programación por librerías. Las diferencias principales entre ellas son que: las librerías *se llaman* desde el código y los frameworks *llaman* al código. En otras palabras, el framework es principalmente un grupo de librerías que sirven como esqueleto para los módulos de las aplicaciones que se diseñen. Hay que puntualizar el cambio que se produce con el tipo de arquitectura utilizada por frameworks (cuando estas llaman al código), ya que el flujo de datos se invierte si la comparamos con la programación tradicional por procedimientos. Lo que se busca es hacer los componentes de alto nivel menos dependientes de sus subsistemas. Los componentes de alto nivel pasan el control a componentes de bajo nivel, y estos son los que deciden como deberían trabajar y cuando responder.

Symfony (Ardissone & Alvarez, 2011), CakePHP (Cake Software Foundation, Inc., 2017) y Zend Framework usan la arquitectura MVC (Model-View-Controller), lo que se traduce como modelado, vista y controlador. Esta arquitectura permite dividir la aplicación en tres grandes capas (Ardissone & Alvarez, 2011), ver figura 3, las cuales son:

- Vista: Todo lo que se refiere a la visualización de la información, el diseño, colores, estilos y la estructura visual en sí de las páginas.
- Modelado: Es el responsable de la conexión a la base de datos y la manipulación de los datos mismos. Esta capa está pensada para trabajar con los datos como así también obtenerlos, pero no mostrarlos, ya que la capa de presentación de datos es la vista.
- Controlador: Su responsabilidad es procesar y mostrar los datos obtenidos por el Modelado; es decir, este último trabaja de intermediario entre los otros dos, encargándose también de la lógica de negocio.

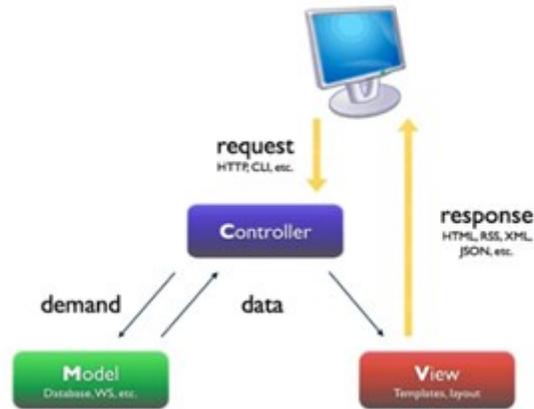


Figura 3. Estructura Modelo-Controlador-Vista (Ardissone & Alvarez, 2011)

Esta arquitectura es confirmada por Sitanggang, Ginanjar, Syukur, Trisminingsih, & Khotimah (2017), quienes exponen, por ejemplo, que mediante el framework CakePHP la actuación de la arquitectura de tres capas (visualización, lógica y datos), trabaja de tal forma que en la que la capa lógica se realizan tareas como la recepción de parámetros de entrada que se envían desde la capa de visualización, además de recuperar datos según estos parámetros desde la capa de datos, procesa y devuelve los datos a la capa de visualización.

Ventajas de usar un framework

Los frameworks son utilizados para:

- Aplicaciones con funcionalidades “estándar”, como loggings, tiendas online, páginas informativas, etc.
- Aplicaciones que pueden escalar muy rápido de una pequeña aplicación a ser usada por miles de personas sin tener la necesidad de cambios radicales en el código.
- Producir APP (aplicación) consecutivas, en que los módulos pueden ser reusados.
- Desarrollo en empresas grandes, con fechas de entrega cortos, personal rotativo y desarrollo a la medida para clientes.
- Aquel que desea ser un desarrollador web profesional. Aprender a usar frameworks no requiere demasiado esfuerzo.

Desventajas de los frameworks

Los frameworks no son muy útiles para:

- Páginas puramente informativas, sin actualizaciones frecuentes del usuario.
- Pequeños proyectos con conectividad limitada a bases de datos, que no se beneficiarían mucho del código generado por los frameworks.

- Proyectos muy complejos y extensos que necesiten lenguajes compilados en lugar de PHP.

Frameworks ORM

El mapeo entre objetos y relaciones (ORM Object-Relational mapping) es una técnica de programación para convertir la lógica de los objetos en lógica relacional utilizando un lenguaje de programación orientado a objetos y la utilización de una base de datos relacional. En los motores de base de datos se dice que cada tabla es una relación, de ahí el nombre de base de datos relacionales, lo que implica que las tablas se encuentran relacionadas entre sí.

En una aplicación orientada a objetos se dice que se trata con objetos y no con tablas. Cuando se agrega un registro a la tabla de personas por ejemplo, en realidad se dice que se agrega un nuevo objeto "Persona". Cuando se dice que un país está relacionado a varias personas, se está diciendo que un objeto "País" contiene una colección de objetos "Persona". Para esto, se crean clases que mapean cada relación de la base de datos y en lugar de interactuar directamente con la base de datos, se interactúa con los objetos y el ORM se encargará de traducir lo necesario para hablar con la base de datos. Con esto se logra una abstracción casi del 100% con relación al motor de base de datos, sin importar cuál sea, ya que hoy en día la mayoría de ellos se encuentran soportados por los ORM.

Factores de calidad ISO 9126

El estándar ISO 9126 se desarrolló con la intención de identificar los atributos clave del software de cómputo, identifica seis atributos clave de la calidad, ver figura 4 (Pressman, 2010):

Funcionalidad. Grado en el que el software satisface las necesidades planteadas según las establecen los atributos siguientes: adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad.

Confiabilidad. Cantidad de tiempo que el software se encuentra disponible para su uso, según lo indican los siguientes atributos: madurez, tolerancia a fallas y recuperación.

Usabilidad. Grado en el que el software es fácil de usar, según lo indican los siguientes subatributos: entendible, aprendible y operable.

Eficiencia. Grado en el que el software emplea óptimamente los recursos del sistema, según lo indican los subatributos siguientes: comportamiento del tiempo y de los recursos.

Facilidad de recibir mantenimiento. Facilidad con la que pueden efectuarse reparaciones al software.

Portabilidad. Facilidad con la que el software puede llevarse de un ambiente a otro según lo indican los siguientes atributos: adaptable, instalable, conformidad y sustituible.



Figura 4. Los criterios de calidad según ISO 9126 (Pressman, 2010)

Evaluación de cada frameworks

Se usaron los parámetros del ISO 9126 para evaluar cada uno de los tres frameworks definidos para este estudio.

Symfony

Funcionalidad. Se observó que Symfony cumple con las necesidades de los usuarios y con su objetivo que es ser una herramienta profesional para desarrollo en PHP. Lo que se puede evidenciar en sus usuarios actuales, sus contribuidores y su crecimiento mensual, ver figura 5. Symfony aporta filtros para manejar problemas de seguridad, desde los más sencillos como autenticar HTTP, hasta los más complejos, por tanto recibe una buena calificación en esta área.

« *Symfony is a set of PHP Components, a Web Application framework, a Philosophy, and a Community – all working together in harmony.* »

Symfony Framework

The leading PHP framework to create websites and web applications. Built on top of the Symfony Components.

Symfony Components

A set of decoupled and reusable components on which the best PHP applications are built, such as Drupal, phpBB, and eZ Publish.

Symfony Community

A huge community of Symfony fans committed to take PHP to the next level.

Symfony Philosophy

Embracing and promoting professionalism, best practices, standardization and interoperability of applications.

Symfony Project stats



Figura 5. Vista del home page de Symfony (SensioLabs, 2017)

Confiabilidad. Se encontraron pocos reportes de fallas de Symfony, y un excelente procedimiento para reportar bugs dentro de su página oficial (Fabien Potencier, 2017). Por lo que Symfony califica no solo como un framework maduro y tolerante a fallas, sino también como un software en constante mejora.

Usabilidad. No es fácil empezar a trabajar en Symfony, su curva de aprendizaje es amplia. SensioLabs ofrece una certificación online: SF3C6, Symfony 3.0 Certification, con una duración de 90 días (SensioLabs, 2017), además dicho curso tiene como prerrequisito 2 cursos básicos. Aprender a usar Symfony toma tiempo.

Eficiencia. En un estudio de comparación de rendimientos de Porebski (2011) usando un procesador Intel i5-750 2.67-GHz con 8GB RAM, se usó JMeter para 10 usuarios haciendo 1,000 consultas. En este estudio usando Symfony obtuvo un resultado de 221 páginas vistas por segundo por medio del servidor. Que fue un resultado mayor al de CakePHP, pero menor al de Zend.

Facilidad de recibir mantenimiento. Symfony provee un software de pruebas llamado Lime. Las pruebas en Lime consisten en llamar a una función o método, ver Tabla 2. Estas funciones tienen entradas predefinidas y luego se compara el resultado con las salidas esperadas. Lime demostró ser un software demasiado básico para su propósito.

Método	Descripción
ok(\$test)	Prueba una condición y pasa si es true
is(\$value1, \$value2)	Compara dos valores y pasa si son iguales (==)

isnt(\$value1, \$value2)	Compara dos valores y pasa si son distintos
like(\$string, \$regexp)	Prueba una cadena contra una expresión regular
unlike(\$string, \$regexp)	Comprueba que la cadena difiera de la expresión regular
is_deeply(\$array1, \$array2)	Comprueba que dos arrays tienen los mismos valores

Tabla 2. Métodos de Lime

Portabilidad. Symfony es multiplataforma, funciona para sistemas operativos Windows y derivados de UNIX; sin embargo solo funciona para versiones de PHP superiores a PHP 5.

CakePHP

Funcionalidad. CakePHP es intuitivo y ayuda a un rápido desarrollo, cumple con ese objetivo, pero tiene falencias en otras áreas. Sus elementos de pruebas son bastante sencillos, al igual que sus propiedades de seguridad y su modelo ORM.

Confiabilidad. El nivel de depuramiento para producción esta inicializado en cero por defecto, lo que podría esconder algunos errores; esto no es una buena práctica, y le quita confiabilidad a CakePHP.

Usabilidad. La mejor característica de CakePHP, sencillez es el adjetivo que mejor lo describe. CakePHP tiene una buena evaluación cuando se trata de reutilización (Li, Karnan, & Chishti, 2017).

Eficiencia. En el estudio de comparación de rendimientos Porebski (2011) anteriormente descrito, CakePHP tuvo un resultado de 178. Es el peor rendimiento de los tres frameworks estudiados.

Facilidad de recibir mantenimiento. Cake tiene una jerarquía y organización que facilitan el mantenimiento, así como su efectividad para limitar la redundancia (Golding, 2008).

Portabilidad. CakePHP además de funcionar en sistemas operativos Windows, Mac y Linux soporta todas las versiones de PHP, por lo que resulta muy portable.

Zend Framework

Funcionalidad. Zend tiene el atributo de ser adaptativo y fácilmente extensible, tiene un gran rendimiento y herramientas para garantizar la seguridad.

Confiabilidad. Zend es un framework maduro, tolerante a fallas.

Eficiencia. Zend obtuvo el mejor resultado en la prueba de comparación de rendimientos (Porebski, 2011).

Facilidad de recibir mantenimiento. Zend puede recibir plugins por medio de la carpeta Zend_plugins de una forma muy sencilla, lo cual lo convierte en extensible y fácil de recibir mantenimiento. Además la documentación en Zend es excelente.

Portabilidad. Zend funciona para plataformas Windows y derivadas de Unix.

Información general de los frameworks

Symfony fue producida en los laboratorios franceses de desarrollo web Sensio Labs por Fabien Potencier (Ardissonne & Alvarez, 2011), primero fue creado para el desarrollo de sus propias aplicaciones, pero en 2005 fue liberado como un proyecto *open source*.

CakePHP empezó en 2005 de la mano del desarrollador web polaco Michal Tatarynowicz. CakePHP es un proyecto totalmente manejado por la comunidad, y desde sus inicios se ha visto influenciado por Ruby on rails. Las características más importantes de CakePHP son su usabilidad y su destaca velocidad de desarrollo.

Zend Framework fue creado por Zend Technologies Ltd., es una de la empresa israelí de software de infraestructura Wide Web con sede en Cupertino, California, EE.UU., cuyo centro de tecnología está en Ramat Gan, Tel Aviv, Israel y oficinas en Francia, Italia y Alemania. Las operaciones de la compañía están centradas en productos relacionados con el desarrollo, despliegue y gestión de aplicaciones PHP basadas en web.

Se detallaron datos generales de los frameworks estudiados en la tabla 3.

	Symfony	CakePHP	Zend Framework
Desarrolladores:	Comunidad SymfonyLabs	Cake Software Foundation	Zend Technologies
Lanzamiento inicial:	22 de octubre de 2005	Abril 2005	3 de Marzo de 2006
Última versión estable:	3.2.5 , 09 de marzo	3.2.8 , 24 de abril de 2018	3.0.01 , 28 de junio
Género:	Framework Web	Framework	Framework
Soporta:	PHP 5.3	PHP 7.1	PHP 7.1
Sistema operativo:	Multiplataforma	Multiplataforma	multiplataforma
Licencia:	Licencia MIT	Licencia MIT	Nueva licencia BSD

Tabla 3. Información general de cada framework

Hello world

Se revisó por revisión documentada las ventajas y desventajas de cada uno de los tres frameworks propuestos para evaluación, pero también fue importante hacerlo de manera práctica, por medio de un ejemplo sencillo de “hello world” en cada una de estas plataformas. El “hello world” consiste en un mensaje por pantalla que suele ser usado como introducción al estudio de un lenguaje de programación, en este caso frameworks, siendo un primer ejercicio típico.

Hello World en Symfony

Se debe dirigir al root en nuestra web y crear un directorio en esa ubicación, por ejemplo /symfony. Luego, abrir la consola, ir hacia el directorio de esa carpeta y crear un proyecto de symfony:

```
$ symfony generate:project symfony
```

Dentro del proyecto, se crea una aplicación. Se usará el nombre de frontend:

```
$ symfony generate:app frontend
```

Se crea el módulo hello introduciendo los siguientes comandos:

```
$ symfony generate:module frontend hello
```

Ahora se ingresa a <http://localhost/symfony/hello> en el navegador. El resultado se presenta en la figura

6.

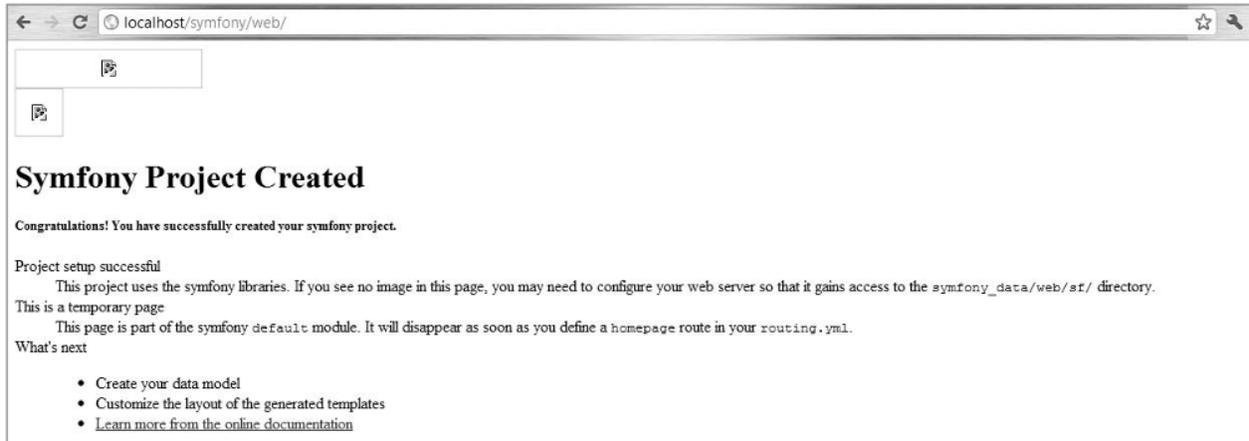


Figura 6. Creación de un módulo en Symfony

A continuación, se debe cambiar la vista normal, por la vista del módulo en uso. Hay que localizar el archivo `/symfony/apps/frontend/config/routing.yml` y cambiar el parámetro `homepage` del módulo `default` al módulo `hello`. Después de editar, se verá así:

`homepage:`

`url: /`

`param: { module: hello, action: index }`

Este código redirige la página principal de nuestra aplicación al módulo en uso. Esto se comprueba en el navegador, pero primero se debe limpiar el cache:

`# symfony cc`

Al revisar la dirección <http://localhost/symfony/hello> se verá una presentación en pantalla similar a la presentada en la Figura 7.



Figura 7. Cambio al módulo hello en Symfony

El módulo hello ha sido creado pero aún se sigue ejecutando el módulo por defecto. Para presentar el módulo creado se debe ir a:

`/symfony/apps/frontend/modules/hello/actions/actions.class.php`

y comentar la función forward de la siguiente forma:

```
<?php class helloActions extends sfActions {
public function executeIndex(sfWebRequest $request) {
//$this->forward('default', 'module');
}
}
```

Ahora si fuese probada en el navegador presentaría una página en blanco porque la vista del índice para el módulo está vacía. Para cambiar esto, ir a:

`/symfony/apps/frontend/modules/hello/templates/indexSuccess.php` y escribir el “hello world”:

```
<h1>Hello World!</h1>
```

Ir a `http://localhost/symfony/hello` y se consigue lo esperado como se muestra en la figura 8.



Figura 8. Hello world en Symfony

Hello World en CakePHP

Primero se crea el controlador; se crea nuevo archivo en `app/controllers` que lleva el nombre de `hello_controller.php`. Los nombres son importantes en CakePHP porque indican el archivo por defecto para el controlador hello. A este controlador se le agrega el siguiente código:

```
<?php class HelloController extends AppController{
function index(){
$this->set( 'test', "Hello World!");
}
}
?>
```

Ahora se crea el módulo. Cuando se establece `$useTable` como `false`, se crea un módulo que no usa una base de datos; que es lo que se necesita para este ejemplo.

```
<?php class Hello extends AppModel{
var $useTable = false;
}
```

?>

Se crea la vista, para presentar la información al usuario. Mientras que en el lado del controlador, se ha creado la variable test que puede ser mostrada en la vista asociada. Además se debe crear una carpeta adicional con el nombre /hello dentro de /app/views. Hay que tener en cuenta que esta vista necesita un archivo de plantilla (template) peculiar de CakePHP extensión .ctp

```
<html>
<body>
<h1><?php echo $test ?></h1>
<p>Congratulations! You have created your first CakePHP project. </p>
</body>
</html>
```

En el archivo app/config/routes.php y se debe comentar los dos esquemas de enrutamiento por default. Y además programar la redirección de la conexión desde la root web hacia el controlador hello y definir index() como la acción por defecto.

```
// Router::connect('/', array('controller' => 'pages', 'action' => 'display', 'home'));
// Router::connect('/pages/*', array('controller' => 'pages', 'action' => 'display'));
Router::connect('/', array('controller' => 'hello', 'action' => 'index'));
```

El resultado final se muestra en la Figura 9.



Figura 9. Hello world en CakePHP

Hello World en Zend Framework

Con la consola se debe ir a la ubicación donde se ubicará la aplicación. Se usó la carpeta root web. Por medio del comando ZF se crea una plantilla de proyecto:

```
$ zf create project zfhello
```

En la dirección web <http://localhost/zfhello/public> se muestre un mensaje como el de la figura 10.



Figura 10. Main page del proyecto en Zend

En este caso los *controladores* y *vista* por defecto ya fueron creados por el comando `zf create Project`, así que no fue necesario codificar nada de estas dos partes; esto se pudo comprobar en el archivo `IndexController.php`:

```
<?php class IndexController extends Zend_Controller_Action {
    public function init() {
        /* Initialize action controller here */
    }
    public function indexAction() {
        //action body
    }
}
```

Este código sirve simplemente como una interfaz, y se puede modificar posteriormente. Por otro lado, el código de la vista que se encuentra en `index.phtml`. *phtml* es un formato pre-HTML que es procesado antes de ser usado como una página HTML. Y también puede tener scripts PHP incrustados. En este ejemplo se agregó el mensaje de hello world.

```
<style>
```

```
//various style definitions
</style>
<div id="welcome">
<h1>Welcome to the <span id="zf-name"> Zend Framework!</span></h1>
<h3>This is your projects main page </h3>
<div id="more-information">
<p><img src=http://framework.zend.com/images/PoweredBy_ZF_4LightBG.png /></p>
<p>
Helpful Links: </br>
<a href=http://framework.zend.com/>Zend Framework Website</a>
<a href=http://framework.zend.com/manual/en/>Zend Framework Manual </a>
</p>
</div>
</div>
```

El resultado esperado se muestra en la Figura 11.



Figura 11. Hello world en Zend Framework.

Ventajas y Desventajas

Symfony

Entre las principales características de Symfony se incluyen una instalación sencilla, soporte de motor de base de datos común, cumplimiento de patrones de diseño, una versión estable e integración con otras bibliotecas (Bandirmali, 2018). Es una excelente herramienta para profesionales. Posee muchas características y sigue muy estrictamente las buenas prácticas de programación.

Ventajas:

- Doctrina ORM integrada.
- Las clases de modelo pueden ser generadas basadas en esquemas o en bases de datos usando consolas. Esto acelera el proceso de desarrollo.
- Tiene muchos Plug-ins disponibles, la mayoría muy bien documentada y dividida por categorías.
- Tiene CMS (sistema de gestión de contenidos disponibles), algunos como plug-ins.
- Muy buenas herramientas de depuración.

Desventajas:

- Algunos plug-ins no están documentados lo que produce problemas.
- La documentación de symfony se puede calificar como buena, mientras que en otros frameworks es excelente.
- Symfony puede ser grandioso cuando se ha prendido a usarlo, pero su curva de aprendizaje es claramente más amplia que la de otros frameworks.

CakePHP

CakePHP es una mezcla de desarrollo rápido y amigable. Es muy intuitivo y provee buenos tutoriales, lo que hace que su uso sea muy fácil, se produce un rápido progreso de aprendizaje incluso para los más principiantes. Este framework soporta: AJAX, computación en la nube, mensajes de errores customizados, personalización y estensibilidad, depuración, validación de formularios, internacionalización, soporta para frameworks javaScript, Mapeo objeto-relacional (ORM), Soporte para Windows, Linux y Os X, soporte para REST, permite Scaffolding, plantillas y marco de pruebas (Salas-Zárate et al., 2015).

Ventajas:

- CakePHP ofrece su propia solución ORM muy buena.
- Ofrece una excelente integración entre modelos y formas sin mucho código.
- Un módulo de probado incluido.
- Una interfaz web para ejecutar las pruebas.
- La clase sanitize que ayuda a filtrar o limpiar las entradas del usuario.
- Muchos tutoriales y ejemplos en <http://bakery.cake.php.org/>
- Buenas soluciones CMS desde las más simples hasta muy avanzadas.

Desventajas:

- Por defecto hay una unión muy fuerte entre los controladores y el modelo; esto es difícil de cambiar cuando se quiere programar un comportamiento a la medida.
- Sigue utilizando PHP4 esperamos que las próximas versiones se actualicen
- Desempeño flojo.

Zend Framework

Zend framework proporciona muchos módulos y clases como resultado del desarrollo a lo largo de los años por parte de la comunidad de código abierto y es fácilmente extensible (Cheong, Chiew, & Yap, 2010). Zend tiene clases muy útiles, pero no se ha ajustado para nada al entorno ORM. Es muy flexible y extensible.

Ventajas:

- Provee algunas funciones para filtrar las entradas del usuario.
- Algunos CMS disponibles.
- Documentación de altísima calidad.
- Excelente desempeño.

Desventajas:

- Al no estar ajustado al entorno ORM, es necesario escribir tres archivos llenos de código para utilizar un modelo.
- No tiene un repositorio de plug-ins.
- Su tamaño en disco es muy grande: 25 MB formado por miles de archivos.

Usar Zend Framework y MVC es un completamente diferente cuando se trata de administrar las conexiones de su base de datos. Zend proporciona una aplicación que funciona como tutorial para comenzar con Zend Framework (Olen, 2010).

Conclusiones

Basados en la revisión bibliográfica y el ejemplo práctico, se concluye que no existe un framework específico que supere en todas las áreas a los otros evaluados. El enfoque de cada uno es diferente, lo que puede evidenciarse en la forma que han sido desarrolladas estas herramientas, por ejemplo, desde un principio CakePHP tuvo la visión de ser una herramienta que aporte velocidad a los tiempos de desarrollo y tal como se evidenció cumple con este requerimiento.

El enfoque de cada herramienta también se evidencia en la ayuda que ofrecen para cumplir con la estructura MVC, por ejemplo, Zend ayuda a crear los módulos “Controlador y Vista” por defecto, mientras que CakePHP por defecto aporta una unión muy fuerte entre los controladores y el modelo. De forma más evidente se percibe este enfoque en la información que despliegan en sus páginas web y como buscan acaparar nuevos desarrolladores que usen sus herramientas.

Queda claro que Symfony no es recomendable para desarrolladores novatos. CakePHP al contrario, es la plataforma recomendable para desarrolladores que quieren empezar a usar frameworks. La unión fuerte entre los controladores y el modelo es buena para desarrollos básicos. Pero puede ser un serio problema si se quiere desarrollar con más control sobre cada componente.

Zend tiene tres puntos fuertes: su desempeño, su documentación y su seguridad. Como la mayoría de las herramientas de desarrollo que tienen una buena performance, Zend está formado por una cantidad grandísima de documentos que se deben descargar para poder instalarlo.

Queda a discreción del desarrollador, teniendo en cuenta su nivel de experiencia, expectativa de crecimiento de sus aplicaciones y aspiración de rendimiento, elegir cual es la plataforma que más se ajusta a sus necesidades. Las herramientas evolucionan y emergen nuevas exponencialmente, futuros trabajos deben continuar evaluando estas herramientas de apoyo a los desarrolladores de software.

Referencias Bibliográficas

Ardissonne, J., & Alvarez, M. (2011). *Curso Symfony 2*. Venezuela: Creative Commos.

Bandirmali, N. (2018). mtCMF: A novel memory table based content management framework for automatic website generation. *Computer Standards & Interfaces*, 58, 43-52.
<https://doi.org/10.1016/j.csi.2017.12.002>

Biermann, M. (2014). A simple versatile solution for collecting multidimensional clinical data based on the CakePHP web application framework. *Computer Methods and Programs in Biomedicine*, 114(1), 70-79. <https://doi.org/10.1016/j.cmpb.2014.01.007>

Cake Software Foundation, Inc. (2017). CakePHP - Build Fast, grow Solid. Recuperado 10 de diciembre de 2017, a partir de <https://cakephp.org/>

Chao, J., Parker, K., & Davey, B. (2013). Navigating the framework jungle for teaching web application development. En *Proceedings of the Informing Science and Information Technology Education Conference*. Informing Science Institute.

Cheong, S. N., Chiew, W. W., & Yap, W. J. (2010). Design and development of Multi-touchable E-restaurant Management System. En *Science and Social Research (CSSR), 2010 International Conference on* (pp. 680–685). IEEE.

Colomina, I., Arnedo-Moreno, J., & Clariso, R. (2013). A Study on Practices against Malware in Free Software Projects (pp. 1070-1075). IEEE. <https://doi.org/10.1109/WAINA.2013.245>

Denoncourt, D. (2012, enero). CoC Frameworks for PHP and Java. *System iNEWS*, 8-14.

Earley, E. (2009). PHP Frameworks: Frameworks can save your development team valuable time and energy. *System iNEWS*, 37-39.

Elhakeem, Y. F. G. M., & Barry, B. I. (2013). Developing a security model to protect websites from cross-site scripting attacks using ZEND framework application. En *Computing, Electrical and Electronics Engineering (ICCEEE), 2013 International Conference on* (pp. 624–629). IEEE.

Fabien Potencier. (2017). Reporting a Bug (Symfony Docs). Recuperado 25 de agosto de 2017, a

partir de <http://symfony.com/doc/current/contributing/code/bugs.html>

Golding, D. (2008). *Beginning PHP*. New York: Apress.

IEEE Spectrum. (2017). Interactive: The Top Programming Languages 2017. Recuperado 18 de agosto de 2017, a partir de <https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2017>

Jarmo\lowicz, J., Zabierowski, W., & Napieralski, A. (2008). Presentation of improvements for PHP programmers, based on Symfony framework. Creation of example portal and description of used technology. En *Modern Problems of Radio Engineering, Telecommunications and Computer Science, 2008 Proceedings of International Conference on* (pp. 595–597). IEEE.

Li, X., Karnan, S., & Chishti, J. A. (2017). An empirical study of three PHP frameworks. En *Systems and Informatics (ICSAI), 2017 4th International Conference on* (pp. 1636–1640). IEEE.

Olen, J. (2010, agosto). Connecting Zend Framework to DB2 for i. *System iNEWS*, 15-23.

Porebski, B. (2011). *Building PHP Applications with Symfony, CakePHP and Zend Framework*. Hoboken, NJ: John Wiley & Sons inc.

Pressman, R. (2010). *Ingeniería de Software*. México, D.F.: Mc Graw Hill.

Rogue Wave Company. (2017). Zend Framework. Recuperado 10 de octubre de 2017, a partir de <https://framework.zend.com/>

Salas-Zárate, M. del P., Alor-Hernández, G., Valencia-García, R., Rodríguez-Mazahua, L., Rodríguez-González, A., & López Cuadrado, J. L. (2015). Analyzing best practices on Web development frameworks: The lift approach. *Science of Computer Programming*, 102, 1-19.
<https://doi.org/10.1016/j.scico.2014.12.004>

Schutt, K., & Balci, O. (2016). Cloud software development platforms: A comparative overview. En *Software Engineering Research, Management and Applications (SERA), 2016 IEEE 14th International Conference on* (pp. 3–13). IEEE.

SensioLabs. (2017a). Symfony 3 Certification Online Coaching with SensioLabs University. Recuperado 10 de octubre de 2017, a partir de <https://training.sensiolabs.com/en/courses/SF3C6-preparation-for-symfony-certification-online-sensiolabs-university>

SensioLabs. (2017b). Symfony, High Performance PHP Framework for Web Development. Recuperado 15 de noviembre de 2017, a partir de <http://symfony.com/>

Sitanggang, I. S., Ginanjar, A. R., Syukur, M., Trisminingsih, R., & Khotimah, H. (2017). Integration of spatial online analytical processing for agricultural commodities with OpenLayers. En *Electrical Engineering and Computer Science (ICECOS), 2017 International Conference on* (pp. 167–170). IEEE.

The PHP Group. (2017). PHP 5 ChangeLog. Recuperado 10 de octubre de 2017, a partir de <http://php.net/ChangeLog-5.php>

Udomkarn, N., & Wakahara, T. (2011). A Seminar Record Editing Algorithm (pp. 434-439). IEEE.
<https://doi.org/10.1109/INCoS.2011.37>