ESTUDIO COMPARATIVO DE MARCOS DE TRABAJO PARA EL DESARROLLO DE APLICACIONES WEB DE CÓDIGO ABIERTO

COMPARATIVE STUDY OF FRAMEWORKS TO THE DEVELOPMENT OF WEB APPLICATIONS OF OPEN SOURCE

AUTOR
CARLOS ANDRÉS GUERRERO ALARCÓN
Magíster en Informática
Unidades Tecnológicas de Santander
Docente Investigador
Líder del Grupo de Investigación en Ingeniería del Software
Diseño y Administración de Sistemas
cinving@uts.edu.co
COLOMBIA

INSTITUCION
Unidades Tecnológicas de Santander UTS
Institución de Educación Superior
Calle de los Estudiantes #9-82 Ciudadela Real de Minas. PBX: 6413000-6413264
Bucaramanga-Santander
COLOMBIA

AUTOR
HERNANDO RECAMAN CHAUX
Magíster en Informática
Politécnico Colombiano Jaime Isaza Cadavid
Docente Planta
Facultad de Ingenierías
Ingeniería Informática
hrecaman@gmail.com
COLOMBIA

INSTITUCIÓN
Politécnico Colombiano Jaime Isaza Cadavid
Universidad
Cra. 48 # 7-151. Conmutador: (574) 319 7900 - 444 7654
Medellín-Antioquia
COLOMBIA

Recepción: Junio 10 de 2009

Aceptación: Octubre 4 de 2009

Temática abarcada por el artículo: Marcos de Trabajo y Desarrollo de Requisitos de Sistemas y Software

Tipo de artículo: Artículo de Invetigación Cientifico y tecnológico

RESUMEN

El trabajo realizado en la formulación y puesta en marcha de diversos proyectos, con soluciones informáticas, y la interacción con diversos equipos de trabajo, arroja un sin número de oportunidades de mejoramiento, una de ellas, es la forma de hacer aplicaciones informáticas. Los marcos de trabajo, tema de este artículo son una rueda importante en este engranaje del proceso de desarrollo software. Por tanto, con el presente artículo se pretende ampliar la visión acerca de las herramientas disponibles en el mercado para construir aplicativos software.

Éste artículo describe parte de los resultados de una investigación enfocada al área de "Frameworks" en español Marcos de Trabajo. Inicialmente se muestra una breve descripción sobre el concepto de Marcos de Trabajo utilizado en este documento, para luego presentar un análisis de los trabajos realizados en ésta misma área, identificando fortalezas y debilidades de cada uno.

El artículo continúa con la presentación de los criterios definidos para la comparación de Marcos de Trabajo; dichos criterios fueron seleccionados con base en el análisis del estado del arte y con el aporte de expertos en el tema.

Se ilustra también el diseño del instrumento definido por los autores, el cual sirve como fuente primaria para la valoración de los Marcos de Trabajo en la investigación, por ende, también las características relevantes de algunos de los Marcos de Trabajo estudiados.

El artículo cierra con la presentación de los resultados cuantitativos obtenidos en cada uno de los criterios analizados, y con base en los datos recopilados para cada criterio propuesto, concluye mostrando cuales podrían llegar a ser los Marcos de Trabajo más viables para ser usados por un equipo de desarrollo.

PALABRAS CLAVES

Caracterización
Marcos de Trabajo
Aplicación Web
Criterios

ABSTRACT

The work carried out in the formulation and beginning of diverse projects, with computer science solutions, and the interaction with diverse work group, throws without number of opportunities of improvement, one of them, is the form to make applications computer science. The frameworks, subject of this paper are an important wheel in this gear of the process of software development. Therefore, with the present paper it is tried to extend the vision about the tools available in the market to construct software applications.

This paper describes part of the results of an investigation focused to the topic of "Frameworks". Initially is showed a brief draw about the concept of framework used in this document, then is presented an analysis of the previous works carried out in this one same topic, identifying strengths and weaknesses of each.

The paper continues with the presentation of defined criteria for the comparison of frameworks; which were selected in base of the analysis of the state of the art and with the contribution of experts in the subject.

The design of the instrument defined by the authors is showed also, which serves like primary source to make valuation of the Frameworks in the research, therefore, relevant issues of some of the frameworks studied as well.

The paper closes with the presentation of the obtained quantitative results in each of the analyzed criteria, and with base in the data collected for each proposed criterion; it concludes showing who could become the most viable frameworks to be used by a developer team.

KEYWORDS

Characterization
Frameworks
Web Application
Criteria

INTRODUCCIÓN

Existen diversos factores que inciden directamente en el desarrollo de las aplicaciones Web, entre ellos dos de los más representativos son: el Tiempo y el Costo. Por un lado, los clientes requieren los desarrollos para ser utilizados a la menor brevedad (tiempo), sin olvidar las exigencias respectivas en cuanto a portabilidad, escalabilidad y funcionalidad (calidad), de igual manera, se espera que dichos desarrollos no sean tan costosos, toda vez que en el mercado existen muchas herramientas que se pueden "reutilizar". Para lograr lo anterior, las empresas de desarrollo han empezado a implementar técnicas y modelos que ofrece la Ingeniería del Software, y han entendido que el camino para un proceso de desarrollo óptimo y con calidad debe estar sustentado en buenas prácticas de diseño y desarrollo, entre esas prácticas sobresalen, la definición y/o selección de una Arquitectura Software acorde al proyecto, un diseño basado en componentes y pensado en la funcionalidad del usuario final, un desarrollo soportado en herramientas y técnicas que permitan acelerar el proceso de desarrollo y por último un modelo de pruebas que sea transversal a toda la construcción del sistema.

Aparecen en el ámbito del desarrollo software los "Marcos de Trabajo", los cuales tienen como premisa soportar los procesos de desarrollo implementando patrones de diseño sin que el desarrollador se entere de cuándo y cómo está usando un patrón determinado, todo esto para buscar la construcción de aplicaciones con el menor esfuerzo posible, y con el lema de no reescribir código. Teniendo en cuenta la importancia de este nuevo elemento dentro del proceso de desarrollo software, se han planteado estudios alrededor de los "Marcos de Trabajo" con miras a tipificarlos, clasificarlos y organizarlos, de tal forma que un equipo de desarrollo pueda en determinado momento seleccionar un Marco de Trabajo de acuerdo a sus necesidades.

Este artículo presenta los estudios comparativos más relevantes sobre "Marcos de Trabajo" en el escenario de código abierto, además, expone las oportunidades de mejoramiento para cada uno de dichos estudios, y propone una comparación más completa, soportado en una variedad de criterios que permiten concluir en la actualidad cual puede llegar a ser la mejor y peor alternativa a la a la luz de los criterios propuestos en esta investigación.

Este artículo describe el concepto de marco de trabajo, su clasificación y características, también se presentan otras caracterizaciones realizadas, ilustrando sus ventajas y desventajas. En el siguiente apartado se definen ocho (8) criterios de comparación y se presenta el instrumento construido para la recolección y análisis de datos. Por último se encuentran las conclusiones resultantes del estudio.

1. MARCO TEÓRICO

1.1 CONCEPTO DE MARCO DE TRABAJO Y SU CLASIFICACIÓN

Un Marco de Trabajo es un conjunto de componentes físicos y lógicos estructurados de tal forma que permiten ser reutilizados en el Diseño y Desarrollo de nuevos Sistemas de Información [1].
Un Marco de Trabajo se puede clasificar según su extensibilidad en:
Caja negra: Conocido como Marco de Trabajo de Arquitectura de Dato-Conducido. Para este Marco de Trabajo no es necesario conocer los detalles internos, pues se utiliza cada elemento a nivel de componentes. Los objetos son creados por medio de secuencias que utilizan y/o envían mensajes a cada clase para implementar el código.
Caja blanca: Conocido como Marco de Trabajo de Arquitectura de Configuración-Conducida. Las instancias de cada clase deben realizarse por medio de la creación de nuevas clases (a través de la herencia) que utilizan la extensión del Marco de Trabajo [2].

1.2 CARACTERÍSTICAS DE LOS MARCOS DE TRABAJO

En la tabla 1 se enuncian las principales características que podemos encontrar en prácticamente todos los Marcos de Trabajo existentes.

TABLA 1. Características identificadas en los Marcos de Trabajo

Características	Descripción
Abstracción de URLs y Sesiones	No es necesario manipular directamente las URLs ni las sesiones, el Marco de Trabajo ya se encarga de hacerlo.
Acceso a Datos	Incluyen las herramientas e interfaces necesarias para integrarse con herramientas de acceso a datos. Disponen de conectores a las bases de datos más conocidas.

Características	Descripción
Controladores	La mayoría de Marcos de Trabajo e interfaces implementa una serie de controladores para gestionar eventos, como una introducción de datos mediante un formulario o el acceso a una página. Estos controladores pueden ser fácilmente adaptables a las necesidades de un proyecto concreto.
Autenticación y control de acceso	Incluye mecanismos para la identificación de usuarios, mediante la solicitud de nombre de usuario y una contraseña cifrada, para restringir el acceso a determinados componentes y usuarios.
Formularios	Cuentan con herramientas para crear formularios, permitiendo la aplicación de controles personalizados a cada uno de los objetos que son utilizados en formularios.
Código Abierto (Open Source)	Reducción de costos en licencias, pues la mayoría de componentes reutilizables son de código abierto.
Reducción de esfuerzo en el desarrollo	Permiten la creación de plantillas para la reutilización de código. Permite la simplificación de problemas comunes en proceso de desarrollo, por ende se optimizan los tiempos de desarrollo.
Solución definida y controlada	Satisface requerimientos claramente definidos en un proyecto.

2. CARACTERIZACIONES DE MARCOS DE TRABAJO EXISTENTES

En el análisis de la literatura del presente trabajo de investigación se encontraron tres (3) fuentes primarias, las cuales hacen parte de la línea de base que sustentan los criterios de comparación definidos y utilizados en este documento. Los estudios están orientados al análisis de herramientas de tipo código abierto [3], sin embargo no contemplan comparaciones con Marcos de Trabajo desarrollados en Java, lo cual se convierte en una oportunidad para mejorar dichos estudios. En el presente trabajo de investigación se fortaleció esta debilidad, por ende se incluyeron en el estudio comparativo, marcos desarrollados para Java, dado que, desde enero de 2007 Sun Microsystems se agregó a la comunidad de código abierto.

Otro aspecto importante en el análisis de la literatura, radica en la inclusión de los criterios “Modelo Vista Controlador” y “Caché” en las comparaciones, aunque dichos criterios tienen fundamentación teórica, en la práctica no generan valor agregado a la hora de obtener resultados en las comparaciones, pues la gran mayoría de los Marcos de Trabajo analizados implementan los requerimientos propuestos por estos dos criterios. A continuación se relacionan los estudios más relevantes realizados a la fecha acerca de la temática en cuestión:

2.1 DENNIS PALLETT

En la comparación de “Dennis” existen dos (2) criterios que hacen referencia al lenguaje de programación, sin embargo se diferencian por la versión del lenguaje, esto indica que en su momento éste estudio contempló la posibilidad de dar soporte a aplicaciones existentes, algo interesante si y solo si las aplicaciones fueron realizadas con un Marco de Trabajo en particular, pues para el caso de nuevos desarrollos no vale la pena realizar esta distinción, pues el lenguaje a utilizar debe hacer referencia a la ultima versión actualizada.

En este estudio se utilizaron los criterios “MVC” y “Caché”, los cuales son muy básicos y prácticamente obligatorios a la hora de realizar comparaciones entre este tipo de herramientas. Se evidenció otra oportunidad de mejoramiento en el área de gestión de bases de datos, pues no especifica un catalogo de posibles opciones de conectividad.

Este estudio contempla el criterio de plantillas para la presentación de la fachada, sin embargo, el criterio es ambiguo y puede solaparse con algunas características del modelo de negocios. A continuación se relacionan los criterios definidos en este estudio:

1. **ZEND - PHP4:** Soporta PHP versión 4.0.
2. **ZEND - PHP5:** Soporta PHP versión 5.0.
3. **MVC:** Indica si la plataforma contempla la división por capas que establece el modelo vista controlador.
4. **Múltiples DB's:** Indica si la plataforma admite bases de datos múltiples sin tener que cambiar nada.
5. **ORM:** Indica si la plataforma admite un "object-record mapper" usualmente una implementación de ActiveRecord. Db Objetos: Indica si la plataforma incluye otros objetos base de datos como un "TableGateWay".
6. **Db Objetos:** Indica si la plataforma incluye otros objetos base de datos como un "TableGateWay".
7. **Templates:** Indica si la plataforma tiene un dispositivo plantilla integrado.
8. **Caching:** Indica si la plataforma incluye un objeto "caching" o algún otro modo de "caching".
9. **Validación:** Indica si la plataforma tiene una validación integrada o componente de filtrado.
10. **AJAX:** Indica si la plataforma llega con soporte integrado para AJAX.
11. **Auth Modulo:** Indica si la plataforma tiene un módulo integrado para manejar autentificación de usuario.
12. **Modules:** Indica si la plataforma tiene otros módulos como un analizador RSS feed, módulo Pdf y otras cosas útiles.

Este estudio fue realizado en el 2006. La tabla 2 presenta el resumen de los marcos de trabajo vs. los criterios de comparación [4].

TABLA 2. Cuadro comparativo del estudio Dennis Pallett

Marco	1	2	3	4	5	6	7	8	9	10	11	12
Zend		✓	✓	✓		✓			✓			✓
Cake	✓	✓	✓	✓		✓			✓	✓	✓	
Symfony		✓	✓	✓	✓	✓		✓	✓	✓	✓	
Seagull	✓	✓	✓	✓		✓	✓	✓	✓		✓	✓
Wact	✓	✓	✓	✓		✓	✓		✓			
Prado		✓		✓			✓		✓	✓	✓	✓
PHP on TRAX		✓	✓	✓	✓	✓			✓	✓		
Zoop	✓	✓	✓	✓		✓	✓		✓	✓	✓	✓
eZ Components		✓		✓		✓		✓	✓			✓
CodeIgniter	✓	✓	✓			✓	✓	✓	✓			✓

Los resultados de este estudio indican que existen tres (3) posibles Marcos de Trabajo que pueden llegar a superar a sus competidores desde la perspectiva de criterios definida, sin embargo los criterios 1 (PHP4) y 8 (caché) no tienen el peso suficiente para descartar o sugerir un Marco de Trabajo.

2.2 FORUM PHP FRAMEWORKS

Este estudio tiene como base los mismos criterios analizados en el apartado anterior, adicionalmente se incluye un criterio denominado "Event Driven Programming" programación basada en gestores de eventos, la cual define qué tanto están ligadas la estructura y la ejecución de los programas con los sucesos que ocurren en el sistema. Este estudio fue realizado en el 2007. Algunas oportunidades de mejora que se evidencian en el estudio son: la separación de los lenguajes de programación; pues se insiste en las versiones; la aplicación de los criterios para el Modelo Vista Controlador y el uso de caché para las consultas del sistema.

El criterio que especifica la gestión y el modelo de acceso a los datos no plantea la selección de múltiples bases de datos, tampoco entrega un catálogo para evidenciar la posible fortaleza del ORM utilizado. En la tabla 3, se presentan los resultados de este estudio [5].

TABLA 3. Cuadro comparativo del estudio Forum PHP Frameworks

Marco	1	2	3	4	5	6	7	8	9	10	11	12	13
Akelos	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Ash.MVC		✓	✓			✓	✓		✓		✓	✓	
CakePHP	✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	
CodeIgniter	✓	✓		✓		✓	✓	✓	✓				
DIY		✓	✓		✓	✓	✓	✓		✓			
eZ Components		✓		✓		✓	✓	✓	✓				
Fusebox	✓	✓	✓	✓				✓		✓		✓	
PHP on TRAX		✓	✓	✓	✓	✓			✓	✓		✓	
PHPDevShell		✓		✓			✓			✓	✓	✓	
PhpOpenbiz		✓	✓	✓	✓	✓	✓		✓	✓	✓		
Prado		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
QPHP	✓	✓	✓	✓		✓	✓		✓	✓	✓	✓	✓
Seagull	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓		
Symfony		✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	
WACT	✓	✓	✓	✓		✓	✓		✓			✓	
WASP		✓	✓			✓	✓		✓	✓	✓	✓	
Zend		✓	✓	✓		✓		✓	✓			✓	
Zoop	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓		

2.3 KUMBIA VS CAKE VS SYMFONY

El estudio en la tabla 4, aunque más acotado en su alcance; pues solo contempla la comparación de tres (3) Marcos de Trabajo; es tal vez, el más detallado de los tres en cuanto a la definición de criterios para la comparación. Su limitante principal radica en la no inclusión de Marcos de Trabajo realizados para Java [6].

TABLA 4. Cuadro comparativo del estudio Kumbia vs Cake vs Symfony

Característica	Kumbia	Symfony	Cake
MVC Nativo	✓	✓	✓
Enrutamiento Avanzado	✓	✓	✓
Scaffold (Generadores de código)	✓	✓	✓
Scaffold Avanzado (Generadores de Formularios)	✓		
Correo Electrónico	✓	✓	✓

Característica	Kumbia	Symfony	Cake
Mapeo Objeto-Relacional	✓	✓	✓
Asociaciones ORM	✓	✓	✓
Eventos ORM	✓	✓	✓
Sistema de Plantillas	✓	✓	✓
Integración con Smarty	✓	✓	✓
Generación de Reportes (Múltiples Formatos)	✓		
HTML Helpers	✓	✓	✓
Plug-Ins (Integración Terceros)	✓	✓	✓
I18n (Internacionalización)		✓	✓
Efectos Visuales	✓	✓	✓
Integración AJAX	✓	✓	✓
Componente de Sesiones	✓	✓	✓
Logres	✓	✓	
Documentación en Inglés		✓	✓
Documentación en Español	✓	✓	
ACL (Access Control Lists)	✓	✓	✓
Soporte MySQL	✓	✓	✓
Soporte PostgreSQL	✓	✓	✓
Soporte Oracle	✓	✓	
Soporte SQL Server		✓	
Soporte SQLite		✓	✓

La estrategia de dividir los criterios en unidades atómicas es interesante, porque aporta mayor información acerca de las características del marco de trabajo. Sin embargo, esto implica costos en el análisis de los datos y disponer de un equipo de personas con amplios conocimientos de desarrollo de aplicaciones Web, por no decir con conocimientos en Arquitectura software, patrones de diseño, diseño y desarrollo de sistemas Web.

3. CRITERIOS PARA LA COMPARACIÓN DE MARCOS DE TRABAJO

La definición e identificación de criterios se lleva a cabo en dos instancias, la primera en el análisis y la evaluación de la literatura recopilada, en donde se identifican y seleccionan los criterios más relevantes, y la segunda con la ayuda y colaboración de expertos en la implementación de aplicaciones utilizando Marcos de Trabajo de amplio reconocimiento[1].

3.1 LENGUAJE

Este criterio fue heredado de los estudios anteriores, sin embargo la forma de evaluarlo cambia sustancialmente, dichos estudios solo analizan la posibilidad de utilizar un solo lenguaje en diferentes versiones, la variación para este criterio es que se puede comparar cualquier lenguaje en cualquier versión, pues su orientación está más ligada a la complejidad del lenguaje en términos de dificultad en su aprendizaje.

1 MSc. Víctor Hugo Cárdenas, Ing. Edwin Logreira, MSc. Luz Elena Gutiérrez, entre otros colaboradores.

3.2 DOCUMENTACIÓN

Este criterio tiene en cuenta dos aspectos: Por un lado la cantidad de documentación que existe para cada uno de los Marcos de Trabajo y por otro lado el nivel de complejidad que involucra cada uno de los módulos que están documentados. El idioma representa un factor preponderante a la hora calificar este criterio, sin embargo solo se contemplan Marcos de Trabajo desarrollados en inglés y en español.

3.3 CURVA DE APRENDIZAJE

Al igual que el criterio anterior, hace parte de los nuevos criterios de comparación. Surge del análisis realizado a los procesos de capacitación que se realizan en las empresas de desarrollo de software consultadas, para analizar este criterio se tienen en cuenta tres (3) niveles de aprendizaje: mínimo, intermedio y superior.

3.4 COMPATIBILIDAD CON BASES DE DATOS

Este criterio también es herencia de los estudios anteriores, sin embargo se incluye en su análisis la cantidad y la oferta (principales motores relacionales) que el Marco de Trabajo puede ofrecer al momento de interactuar con la capa de acceso a datos.

3.5 VALIDACIÓN DE DATOS EN EL CLIENTE

Las herramientas de validación hacen referencia a las verificaciones que se realizan en las capturas de datos antes de enviar peticiones al servidor. Aquellas verificaciones realizadas como parte del modelo de negocios no hacen parte de este criterio de hecho se contemplan para este criterio el uso de códigos reutilizables en JavaScript y el uso controles finales de usuario.

3.6 SOPORTE PARA AJAX

Este criterio hace referencia a la disponibilidad de componentes que realizan interacción con el servidor, sin necesidad que el cliente visualice cambios de interfaz. También analiza la forma de implementar el desarrollo de componentes de este tipo y la facilidad para integrar el trabajo realizado con AJAX, se cualifica si es práctica o por el contrario si tiene alto nivel de complejidad [7].

3.7 MÓDULOS PARA PRESENTACIÓN DE DATOS

La presentación de informes y salida de formación formateada es de vital importancia, este criterio hace parte de los sugeridos por los expertos en desarrollo, y se mide por la cantidad y variedad de formatos con el cual se puede exportar información (formatos xls, pdf, rtf, entre otros).

3.8 UTILIDADES GRÁFICAS

Hace parte de las observaciones a nivel de presentación realizadas por los expertos en desarrollo consultados, este criterio tiene en cuenta la cantidad de utilidades visuales que contienen cada uno de los marcos de trabajo, tales como: ventanas flotantes, mensajes en colores, alertas, ayudas visuales. Estos desarrollos facilitan la utilización del marco de trabajo, tanto para el programador como para el cliente.

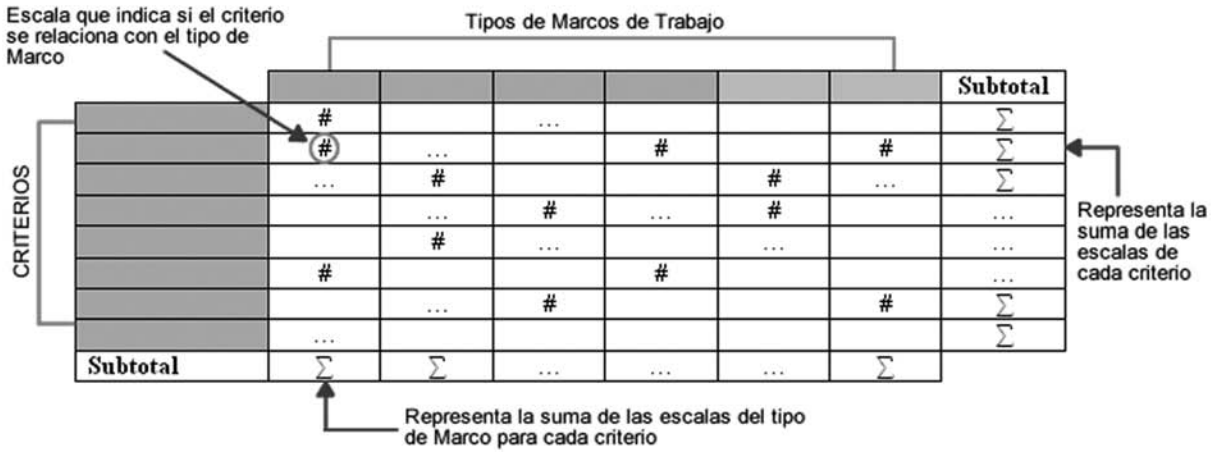
4. ESTUDIO COMPARATIVO

4.1 CONSTRUCCIÓN DEL INSTRUMENTO

La comparación de los Marcos de Trabajo tiene como base los criterios definidos y seleccionados en este trabajo de investigación, por ende, fue necesario diseñar y crear un instrumento que permitiera recopilar la información encontrada. Dicho instrumento consiste en una matriz de pesos, la cual permite verificar cuantitativamente y de manera visual cuál de los Marcos de Trabajo tuvo un mejor desempeño en las pruebas de campo.

A nivel horizontal se encuentran los criterios que conforman la base para la comparación y a nivel vertical se encuentran los Marcos de Trabajo que fueron parte del objeto de estudio. Una vez agrupada la información se debe analizar cada uno de los criterios en cada Marco de Trabajo, lo que implicó disponer de cada uno de ellos para construir un prototipo que permitiera emitir juicios de valor sobre los criterios analizados. La figura 1 presenta el esquema general del instrumento diseñado y utilizado.

FIGURA 1. Esquema de convenciones del instrumento



Cada intersección entre fila y columna debe llevar un valor que permita cuantificar el posible resultado del análisis realizado a un criterio, éstos valores pueden ser sumados a nivel horizontal y equivalen a los totales obtenidos en la evaluación de un criterio particular, de esta forma se puede observar rápidamente cual criterio fue con mayor puntaje, lo cual podría llegar a descartarlo dependiendo del análisis que se realice. De igual forma la sumatoria de las columnas corresponde al valor obtenido en la calificación de los Marcos de Trabajo, este valor indica cual de todos tuvo el mejor desempeño en la evaluación realizada.

Para las intersecciones se planteó un modelo de escala a tres valores, en donde el mínimo valor equivale a Uno (1) y representa el valor más pesimista o la no presencia de una cualidad o característica evaluada, el mayor valor equivale a Tres (3) representa el valor optimista, o la presencia y cumplimiento de todo lo especificado en el criterio evaluado. El valor intermedio es Dos (2), el cual representa un punto de control asignado si y solo si el criterio evaluado no cumple con las otras dos medidas (cada criterio analizado tiene su respectivo análisis y la forma de identificar este valor rápidamente). La tabla 5 resume las convenciones propuestas para la medición.

TABLA 5. Convenciones de la escala del instrumento

Valores	Descripción
Tres (3)	Mayor valor positivo
Dos (2)	Punto de control asignado por la no identificación de los otros dos valores.
Uno (1)	Mínimo valor, que normalmente indicará ausencia del criterio

4.2 APLICACIÓN DEL INSTRUMENTO

4.2.1 Paso 1: Identificación de Marcos de Trabajo analizados

Parte de la estrategia de este trabajo, por mejorar los resultados de sus trabajos predecesores, fue la inclusión de Marcos de Trabajo realizados para Java, pues estos también son Open Source. Otra estrategia fue seleccionar Marcos de Trabajo utilizados como fuente de estudio en otros estudios similares, de esta manera de estableció una masa crítica aceptable que permitiera emitir juicios de valor al final del trabajo realizado.

La tabla 6 presenta el listado de los Marcos de Trabajo objeto del presente estudio.

TABLA 6. Listado de Marcos de Trabajo analizados

Marco	URL
Cake	http://www.cakephp.org/
Akelos	http://www.akelos.org/
Pear	http://pear.php.net/
Symfony	http://www.symfony-project.com/
Zend	http://framework.zend.com/
Ruby On Rails	http://www.rubyonrails.org.es/
Grails	http://grails.org/
Zoop	http://zoopframework.com/
CodeIgniter	http://codeigniter.com/
Kumbia	http://www.kumbiaphp.com
Struts	http://struts.apache.org
Spring	http://www.springframework.org

4.2.2 Paso 2: Análisis individual por criterios

Lenguaje

Para el caso de aquellos marcos de trabajo que usan C, C++, o lenguajes directamente heredados como base para sus desarrollos, tienen una calificación de tres (3) puntos, mientras que aquellos que tienen un nivel de complejidad un poco mayor tendrán una calificación de dos (2) puntos, aquellos que son poco conocidos, de poca aplicación independientemente de su complejidad tienen una calificación de uno (1) punto. Ver tabla 7.

TABLA 7. Análisis del criterio Lenguaje

Marco	Descripción	Escala
Cake	PHP4, PHP5	3
Akelos	PHP4, PHP5	3
Pear [8]	PHP5	3
Symfony	PHP5	3
Zend	PHP5	3
Ruby On Rails	Ruby	1
Grails	Groovy [9]	1
Zoop	PHP4, PHP5	3
CodeIgniter	PHP4, PHP5	3
Kumbia [10]	PHP5	3
Struts [11]	Java	2
Spring	Java	2

Documentación

En la tabla 8, un Marco con documentación, práctica, sencilla, puntual y completa obtiene una calificación de tres (3) puntos, si contiene documentación, pero es confusa y hace referencia a repositorios de difícil acceso tiene una calificación de dos (2) puntos, si no tiene documentación ó la que tiene es deficiente para la comprensión del Marco de Trabajo, tiene una calificación de uno (1) punto.

TABLA 8. Análisis del criterio Documentación

Marco	Descripción	Escala
Cake	Practica, sencilla, puntual, completa	3
Akelos	Practica, sencilla, deficiente	2
Pear	Documentación robusta, deficiente	2
Symfony	Practica, sencilla, puntual, completa	3
Zend	Practica, sencilla, completa	3
Ruby On Rails	Practica completa,	2
Grails	Practica, sencilla, deficiente	1
Zoop	Practica, sencilla,	2
CodeIgniter	Practica, sencilla, completa	2
Kumbia	Practica, sencilla, puntual, completa	3
Struts	Deficiente	1
Spring	Practica, sencilla, completa	3

Curva de aprendizaje

Se considera un Marco de Trabajo con nivel de aprendizaje mínimo aquel que no requiere mucho tiempo de comprensión y tiene una documentación práctica, sencilla puntual y completa, obteniendo una calificación de tres (3) puntos. Si tiene un nivel de aprendizaje intermedio y se requiere demasiado tiempo por contar con una documentación incompleta o confusa tiene calificación de dos (2) puntos. Si tiene un nivel de aprendizaje muy extenso en el tiempo por no tener la documentación, tiene una calificación de uno (1) punto. Ver tabla 9.

TABLA 9. Análisis del criterio Curva de aprendizaje

Marco	Descripción	Escala
Cake	Nivel Intermedio	2
Akelos	Nivel Intermedio	2
Pear	Nivel Intermedio	1
Symfony	Nivel Mínimo	3
Zend	Nivel Intermedio	2
Ruby On Rails	Nivel Intermedio	2
Grails	Nivel Superior	1
Zoop	Nivel Intermedio	2
CodeIgniter	Nivel Superior	1
Kumbia	Nivel Mínimo	3
Struts	Nivel Superior	1
Spring	Nivel Intermedio	2

Compatibilidad con bases de datos

En la tabla 10, los Marcos de Trabajo que sean compatibles con tres o más bases de datos tienen una calificación de tres (3) puntos, mientras que aquellos que tienen compatibilidad con dos bases de datos tendrán una calificación de dos (2) puntos y aquellos que se limitan a una sola base de datos tienen una calificación de uno (1) punto.

TABLA 10. Análisis del criterio Compatibilidad bases de datos

Marco	Descripción	Escala
Cake	MySQL-PostgreSQL-SQLite	2
Akelos	MySQL, PostgreSQL, SQLite	2
Pear	SQLite y Sybase. Mysql, mysqli, oci8, odbc, pgsql, Mysql, mysqli, oci8, odbc, pgsql,	3
Symfony	MySQL, PostgresSQL, Oracle, SQL Server, SQLLite	2
Zend	IBM DB2, MySQL, Oracle, Microsoft SQL Server, PostgreSQL, SQLite	2
Ruby On Rails	MySQL, PostgreSQL, Oracle, SQLServer	2
Grails	JAVA, Oracle	2
Zoop	MySQL, PostgreSQL, Oracle	2
CodeIgniter	MySQL, MySQLi, MS SQL, Postgre, Oracle, SQLite, y ODBC	2
Kumbia	MySQL, PostgreSQL, Oracle	2
Struts	MySQL, Oracle	2
Spring	PostgreSQL, MySql	2

Validación de datos en el cliente

En la tabla 11, los Marcos de Trabajo que tengan validaciones incluidas, con tres o más herramientas tienen una calificación de tres (3) puntos, mientras que aquellos que incluyan dos herramientas tienen una calificación de dos (2) puntos y aquellos que no tengan validación incluidas obtienen una calificación de un (1) punto.

TABLA 11. Análisis del criterio Validación datos

Marco	Descripción	Escala
Cake	Alfanumérica, e-mail, máximo de caracteres, URL y teléfono, entre otras.	3
Akelos	El XhtmlValidator puede ser utilizado para validar los documentos XHTML. It uses only the expat extension functions always available under since PHP 3 and it does not need other external XML processing extensions.	1
Pear	QuickForm guarda automáticamente los valores fijados para elementos a través del envío del formulario, muestra mensajes de error, permite la validación y la filtración que se puede aplicar a los campos individuales y/o al formulario completo y genera el código JavaScript para la validación en el lado cliente.	3
Symfony	Reglas de validación de un formulario, mediante el uso de archivos YAML. Validador de cadenas de texto. Validador de números.	2
Zend	Zend_Filter_Input provides a declarative interface to associate multiple filters and validators, apply them to collections of data, and to retrieve input values after they have been processed by the filters and validators.Zend_Filter_Input proporciona una interfaz de múltiples filtros y validadores, aplicables a las colecciones de datos, y para recuperar los valores de entrada después de que han sido procesados por los filtros y validadores. Values are returned in escaped format by default for safe HTMLThe Zend Framework comes with a standard set of validation classes, which are ready for you to use.Viene con un conjunto estándar de validación de las clases, que están listos para ser utilizados. Funciones incorporadas para validar fechas y correos.	2
Ruby On Rails	Validación de datos de formularios o formas, templates para aplicaciones, recibir y enviar correo electrónico, formato y manipulación de fecha y hora, sesiones y manejo de cookies.	2
Grails	Allows rendering of errors in different formats (at the moment only an HTML list is implemented)Permite la presentación de los errores en diferentes formatos. It is often useful to highlight using a red box or some indicator when a field has been incorrectly input. Coloca en alto relieve mediante un cuadro rojo o algún indicador cuando ha sido un campo de entrada incorrecta. This can also be done with the by invoking it as a method.	2
Zoop	Aviso del JavaScript, etapas de validación a través de la forma y le muestra cada una de las partes que está mal. El formulario también se acuerda de los valores que entraron de manera que los usuarios no tienen que entrar de nuevo.	2
CodeIgniter	No posee validación integrada.	1

Marco	Descripción	Escala
Kumbia	Generación Inmediata de Formularios CRUD (Create, Read, Update, Delete) sobre Entidades de la base de datos. Validación Automática de Tipos de Datos (Numéricos, Texto, Fechas, e-mail y Tiempo). Validación de Integridad Relacional (Llaves Únicas, Llaves Foráneas y Valores de Dominio).	3
Struts	Las solicitudes del cliente se transmiten a un servlet denominado ActionServlet, que actúa como controlador.	2
Spring	HDIV ayuda a automatizar las validaciones de entrada.	2

Soporte para AJAX

Para los Marcos de Trabajo con facilidad para la integración con AJAX se asigna una calificación de tres (3) puntos, mientras que aquellos que tienen una integración difícil tienen una calificación de dos (2) puntos y aquellos que no dispongan de la integración con AJAX reciben una calificación de uno (1) punto. Ver tabla 12.

TABLA 12. Análisis del criterio Soporte AJAX

Marco	Descripción	Escala
Cake	Integración fácil, practica	3
Akelos	Integración fácil	3
Pear	Integración practica	2
Symfony	Integración fácil, practica	3
Zend	Integración practica	2
Ruby On Rails	Integración con AJAX ,fácil	3
Grails	Integración, fácil, practica,	3
Zoop	Integración fácil, practica	3
CodeIgniter	Integración practica	2
Kumbia	Integración fácil, practica	3
Struts	Integración practica	2
Spring	Integración practica	2

Módulos para presentación de datos

En la tabla 13 se describen los Marcos de Trabajo que generen reportes en tres formatos tienen una calificación de tres (3) puntos, mientras que aquellos que generen reportes en dos formatos obtendrán una calificación de dos (2) puntos y aquellos que se limitan a un solo formato reciben una calificación de un (1) punto.

TABLA 13. Análisis del criterio Módulos presentación datos

Marco	Descripción	Escala
Cake	pdf, xls, rtf	3
Akelos	pdf, xls	2
Pear	xls, pdf	2
Symfony	xls, pdf, rtf	3
Zend	pdf, xls, rtf	3
Ruby On Rails	pdf, xls	2
Grails	--	1
Zoop	pdf	1
CodeIgniter	--	1
Kumbia	pdf, xls, rtf	3
Struts	pdf, xls, rtf	3
Spring	pdf, xls	2

Utilidades gráficas

En la tabla 14, un Marco con varias utilidades visuales prácticas obtiene una calificación de tres (3) puntos. Si contiene utilidades visuales, pero no son muy prácticas recibe una calificación de dos (2) puntos, si no tiene utilidades gráficas o las que tiene son deficientes para la utilización adecuada del Marco de Trabajo, se le registra una calificación de uno (1) punto.

TABLA 14. Análisis del criterio Utilidades gráficas

Marco	Descripción	Escala
Cake	layouts, elements, mensajes, botones. Utilidades para diseñar un blog al estilo Web 2.0. Scriptaculous para AJAX y Efectos Visuales.	3
Akelos	Librería script.aculo.us	3
Pear	Librería script.aculo.us	3
Symfony	Librería script.aculo.us	3
Zend	Zend_Controller incorpora una nueva funcionalidad llamada ViewRenderer, que facilita el diseño de las acciones en los controladores. Zend_Gdata cambia su formato por uno más orientado a objetos.	2
Ruby On Rails	Librería script.aculo.us	3
Grails	Soporta librerías de integración con AJAX como RICO.	3
Zoop	Zoop soporta librerías de integración de AJAX como DOGO.	3
CodeIgniter	Librería script.aculo.us	3
Kumbia	Librería script.aculo.us	3
Struts	Layouts [12] Manejo de mensajes de errores Botones flexibles. HTML Tags	2
Spring	Scriptaculous	3

4.2.3 Paso 3: Instrumento aplicado

Luego de finalizar el paso 1, donde se presenta el instrumento a utilizar y se evidencia su utilidad para la recolección de datos, se procede a analizar cada criterio y a cualificar cada Marco de Trabajo, la conclusión de este trabajo realizado se presenta en la figura 2.

FIGURA 2. Instrumento aplicado

	Cake	Akelos	Pear	Symphony	Zend	Ruby On Rails	Grails	Zoop	CodeIgniter	Kumbia	Struts	Spring	Subtotal	
Lenguaje	3	3	3	3	3	1	1	3	3	3	2	2	30	Promedio Filas = 27.5
Documentación	3	2	2	3	3	2	1	2	2	3	1	3	27	
Curva Aprendizaje	2	2	1	3	2	2	1	2	1	3	1	2	22	
Base de Datos	2	2	3	2	2	2	2	2	2	2	2	2	25	
Validación integrada	3	1	3	2	2	2	2	2	1	3	2	2	25	
Integración con Ajax	3	3	2	3	2	3	3	3	2	3	2	2	31	
Módulos RSS, feed, pdf	3	2	2	3	3	2	1	1	1	3	3	2	26	
Utilidades graficas	3	3	3	3	2	3	3	3	3	3	2	3	34	
Subtotal	**22**	**18**	**19**	**22**	**19**	**17**	**14**	**18**	**15**	**23**	**15**	**18**	**220**	
	Promedio Columnas = 18.3													

Análisis a nivel de columnas

De la figura 2, la suma de las columnas indica si la clasificación posee en algún grado los criterios de cada una de las filas. La suma de los totales encontrados en cada columna es 220 y el promedio de esa sumatoria es 18.3. Los valores inferiores a este promedio no son relevantes para el presente trabajo, pues no cuentan con la mayoría de los criterios definidos.

Análisis a nivel de filas

Las filas del instrumento de la figura 2 representan los criterios descritos en la sección "3. Criterios para la comparación de Marcos de Trabajo", y la suma de las filas indican si el criterio está presente en algún grado en las columnas. La suma de los totales encontrados en cada fila es 220 y el promedio de esa sumatoria es 27.5. Los valores superiores al promedio indican cuales son los criterios que más se repiten en los tipos de Marcos de Trabajo.

4.3 RESULTADOS DEL ESTUDIO COMPARATIVO

4.3.1 Criterios más comunes

Se utilizan ocho (8) criterios principales y como resultado de la aplicación del instrumento propuesto en este trabajo, de los cuales tres (3) son comunes a la mayoría de los Marcos de Trabajo analizados. La fundamentación conceptual para rechazar algunos de los criterios obedece al "Análisis a nivel de filas". La tabla 15 presenta los criterios con mayor calificación en el estudio comparativo para la presente caracterización.

TABLA 15. Criterios con mayor calificación

	Puntaje
Lenguaje	30
Integración con AJAX	31
Utilidades gráficas	34

4.3.2 Tipos de Marcos de Trabajo no recomendados

Los tipos de Marcos de Trabajo con la menor puntuación se descartan, puesto que luego de aplicar el instrumento se deduce que no cuentan con las condiciones mínimas para proporcionar a un desarrollador las utilidades de un Marco de Trabajo poderoso y eficiente. La tabla 16 presenta los tipos eliminados.

TABLA 16. Marcos de trabajo descartados

	Puntaje
Grails	14
Codeigniter	15
Struts	15
Ruby On Rails	17

4.3.3 Tipos de Marcos de Trabajo con mejor calificación

La agrupación inicial de Marcos de Trabajo encontrada, está conformada por doce (12) subgrupos, sin embargo, luego de realizar el análisis del instrumento aplicado, dicha clasificación se reduce a tres (3) subgrupos, los cuales hacen parte de la caracterización encontrada. La fundamentación conceptual para rechazar algunos de los subgrupos obedece al "Análisis a nivel de columnas". La tabla 17 presenta las agrupaciones finales.

TABLA 17. Marcos de Trabajo clasificados

	Puntaje
Cake	22
Symfony	22
Kumbia	23

De los anteriores resultados se puede concluir que:

- Los Marcos de Trabajo que ofrecen demasiadas dificultades en el aprendizaje y que obtuvieron una baja evaluación (por lo tanto no los recomendados) son: Grails, Codeigniter y Struts.
- El Marco de Trabajo con el menor puntaje acumulativo en todos los criterios es Grails.
- Los Marcos de Trabajo recomendados por cumplir con la totalidad de criterios en una forma satisfactoria son: Kumbia, Cake y Symfony.
- El Marco de Trabajo con el mayor puntaje acumulativo en todos los criterios es Kumbia.

5. CONCLUSIONES

Las bases conceptuales de este artículo tienen su origen en tres (3) estudios comparativos que fueron la línea de base para el presente trabajo de investigación, adicionalmente se soporta en el trabajo realizado por expertos en el desarrollo de aplicaciones con los Marcos de Trabajo que hacen parte del objeto de estudio, soportado en estás fuentes primarias y en los resultados obtenidos se puede concluir que:

- Los criterios de comparación más comunes de los marcos de trabajo son Lenguaje, Integración con Ajax y Utilidades gráficas. Estos criterios no son significativos para una adecuada clasificación.
- Los marcos de trabajo Grails, Codeigniter, Struts y Ruby On Rails no cuentan con las condiciones mínimas para proporcionar a un desarrollador las utilidades de marcos de trabajo eficientes.
- Los marcos de trabajo recomendados en el presente estudio son: Cake, Symfony y Kumbia, puesto que cuentan con la mayoría de utilidades y características para proporcionar herramientas adecuadas de trabajo.
- Sin embargo, los Marcos de Trabajo más completos y profesionales son los menos eficientes a la hora de desarrollar sistemas para la Web, su nivel de encapsulamiento siendo ejemplar requiere horas y días de trabajo para su adecuación en otros sistemas.
- Un Marco de Trabajo competitivo es aquel que se crea y desarrolla de acuerdo a las necesidades del equipo de desarrollo, lo ideal es tomar como referencia uno de base y adecuar el mismo Marco de Trabajo, de esta manera se obtuvieron resultados interesantes en este proyecto.
- La selección de un Marco de Trabajo no debe ser un proceso taxativo, pueden existir alternativas mejores que otras pero de ninguna manera se puede decir que uno solo es el mejor que todos.
- La construcción y adecuación de Marcos de Trabajo por parte de un equipo de desarrollo, conduce a la innovación y generación de nuevo conocimiento, lo cual en términos de desarrollo software, casi siempre implica menos costos y menos tiempos de producción.

6. REFERENCIAS

[1] MINETTO, Elton Luís. Frameworks para Desenvolvimento em PHP. Brasil: Novatec, 2007; p: 17-18.

[2] MARKIEWICZ, Marcus Eduardo y J.P. DE LUCENA, Carlos. El Desarrollo del Framework Orientado al Objeto. [Consultado noviembre de 2008]. Disponible en: < http://www.lawebdejm.com/prog/uml/Framework.pdf>

[3] MINERA, Francisco. Manual del Programador PHP 5: Evolución y Madurez. Argentina: MP Ediciones, 2006; p: 10-15.

[4] PALLETT, Dennis. Gráfica Comparación Frameworks. [Consultado noviembre de 2008]. Disponible en: <http://lnx.googlewrite.com/chart.php>

[5] PHP Frameworks. [Consultado noviembre de 2008]. Disponible en: <http://phpframeworks.com/index.php>

[6] Andrés Felipe. Kumbia vs. Cake vs. Symfony. [Consultado noviembre de 2008]. Disponible en: <http://minombreesfelipe.blogspot.com/>

[7] PERRY, Bruce W. Ajax. Los Mejores Trucos. Madrid: Anaya Multimedia, 2006; p: 20-45.

[8] COGGESHALL, John. La Biblia de PHP5. Madrid: Anaya Multimedia, Edición 1ª, Mayo 2005; p: 180-190.

[9] JUDD, Christopher M.; FAISAL NUSAIRAT, Joseph y SHINGLER, Jim. *Beginning Groovy and Grails: From Novice to Professional*. USA: Apress, 2008; p: 1-9.

[10] Kumbia PHP Framework. [Consultado diciembre de 2008]. Disponible en: <http://www.kumbiaphp.com/blog/about/>

[11] SPIELMAN, Sue. The Struts Framework: Practical Guide for Java Programmers. USA: Elsevier Inc., 2003.

[12] CEBALLOS, Francisco Javier. *Java 2.* Interfaces Gráficas Y Aplicaciones Para Internet, *2ª Edición.* Madrid: Alfaomega Grupo editor, 2006; p: 10-20.