Medición de atributos POO en frameworks de desarrollo PHP

Julio Acosta[1], Cristina Greiner[1], Gladys Dapozo[1]; Marcelo Estayno[2]

1 Departamento de Informática. Facultad de Ciencias Exactas y Naturales y Agrimensura
Universidad Nacional del Nordeste, Av.Libertad 5450, 3400, Corrientes, Argentina
{gndapozo,cgreiner}@exa.unne.edu.ar; julio_acosta_01@hotmail.com
2 Departamento de Informática. Facultad de Ingeniería. Universidad Nacional de Lomas de Zamora, Ruta 4 Km 2, 1832 Lomas de Zamora, Buenos Aires, Argentina
mestayno@gmail.com.ar

Resumen. La medición es un factor clave para analizar la calidad dado que permite la generación de indicadores cuantitativos sobre una entidad software. En este trabajo se presenta una metodología de medición de atributos característicos de la programación orientada a objetos (POO) sobre aplicaciones escritas en PHP 5, con el objetivo de conocer qué tan bien se implementan los conceptos de la POO en aplicaciones PHP. La misma fue aplicada para evaluar cuatro frameworks de desarrollo PHP, que soportan POO, utilizando la herramienta PHPDepend. En la actualidad los frameworks son ampliamente usados por los desarrolladores dado que permiten agilizar la programación mediante estructuras genéricas configurables, por tanto, utilizar un framework que cumple criterios de calidad, contribuirá a la calidad del producto que se desarrolla. Del análisis de los valores resultantes se obtuvo un ranking de los frameworks que mejor se adecuan a los valores recomendados para las métricas analizadas.

Palabras clave: Medición de atributos de calidad, Métricas Orientadas a Objetos, Frameworks de desarrollo, Lenguaje PHP.

1 Introducción

Los modelos de evaluación y mejora de procesos, tales como: CMM, CMMI, ISO 15504 SPICE, incorporan en sus primeros niveles, como parte de las buenas prácticas recomendadas, técnicas y procesos para el aseguramiento de la calidad que se corresponden con la medición de software, los procesos de revisión y auditoría y las pruebas de software [1].

La medición de atributos de calidad del software representa una ventaja estratégica para las empresas de Software y Servicios Informáticos (SSI), dado que proporciona un mayor conocimiento acerca de los procesos productivos. Medir es conocer, y este conocimiento permite modificar aquellos factores que aportan una mayor eficacia en el proceso productivo, obteniendo productos con un nivel de calidad mayor haciendo a las organizaciones más eficientes y permitiendo una ventaja estructural frente a sus competidores [2].

Las métricas técnicas facilitan una base para que el análisis, diseño, codificación y prueba puedan ser conducidos más objetivamente y valorados más cuantitativamente [3] y permiten determinar mediante estadísticas basadas en la experiencia, la calidad del software según el cumplimiento de parámetros requeridos.

Por otra parte, es creciente la tendencia de la industria de software hacia la adopción del paradigma de la programación orientada a objetos debido a la promoción de características deseables en el software, tales como la reutilización de código, encapsulación, abstracción y modularización, entre otros [4]. Según un estudio reciente sobre tendencias de lenguajes de programación [5], el 57% de los lenguajes más populares se encuentran en la categoría de la POO.

Las métricas OO (Orientación a Objetos) miden los atributos que caracterizan a la POO y aportan información acerca de qué tan bien se aplican los principales conceptos de este paradigma [6].

1.1. Lenguajes de programación

Según el índice de la Comunidad de Programación TIOBE [7], que indica la popularidad de los lenguajes de programación, entre los siete primeros lenguajes más utilizados se encuentran C, Java, Objective-C, C++, C#, Visual Basic y PHP.

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. A partir de PHP 5 incorpora un modelo de objetos más completo. Entre sus características están la inclusión de la visibilidad, las clases abstractas y clases y métodos finales, manejo de excepciones, interfaces, clonación y tipos sugeridos [8].

Forma parte de las tecnologías LAMP (Linux Apache, MySQL, PHP/Python/Perl), siendo esta plataforma una de las más extendidas en cuanto software open source para desarrollo web se refiere [9].

1.2. Herramientas de medición

Las herramientas de medición facilitan la tarea de mantener un mínimo de calidad, contribuyendo con información que permite detectar posibles deficiencias en la construcción del software.

Existe una amplia variedad de herramientas de software enfocadas en la medición de atributos de calidad de aplicaciones OO, pero en su mayoría consideran código Java. No se encontraron muchas herramientas de medición de código PHP, probablemente debido a la aparición más reciente de la versión orientada a objetos de este lenguaje (PHP 5). Entre las herramientas disponibles se puede mencionar:

1.2.1. PHPMD (PHP Mess Detector)

Según la propia página del proyecto [10], esta herramienta es un "spin off" (derivado) de PHP Depend. Es muy similar a la herramienta PMD para Java. Permite detectar código no utilizado, nombres de variables poco apropiados, etc., en base a un conjunto de reglas. Incluye cuatro juegos de reglas [11]:

- Código no utilizado: Localiza métodos que nunca son invocados.
- Tamaño del código: Complejidad ciclomática, longitud de métodos o clases, etc.
- Diseño: Comprueba el grado de acoplamiento, la profundidad de la herencia, etc.
- Nombrado: indica si se han cometido violaciones por usar nombres de variable demasiado cortos o largos, y otras convenciones de nombre.

1.2.2. Sonar

Es una plataforma de código abierto que sirve para gestionar la calidad del código. Cubre diversos aspectos de la calidad del código, tales como Arquitectura y Diseño, Escasez de Comentarios, Duplicaciones de código, Reglas de Código, Errores potenciales, Complejidad, Pruebas Unitarias. Muestra de manera gráfica los valores de las métricas. Utiliza la herramienta Maven para mostrar los resultados del análisis. Inicialmente permitía el análisis de código Java solamente, luego se extendió para cubrir otros lenguajes como PHP, Flex, PL/SQL o Visual Basic, C#, entre otros [12].

1.2.3. PHP Depend

Es un software liviano que realiza análisis de código estático. Para ello toma el código fuente y genera una estructura de datos interna fácilmente procesable, denominada AST (Árbol de Sintaxis Abstracta), que representa las diferentes sentencias y elementos utilizados en el código fuente del proyecto analizado [13]. Mide y reporta valores de las métricas de software que representan diferentes aspectos de calidad. Genera informes que son el resultado del proceso de medición: 1*) Overview Pyramid (OP),* el cual agrupa mediciones inherentes a la herencia, acoplamiento, tamaño y complejidad; 2) *Abstraction Inestability Chart (AI),* el cual establece una relación entre la abstracción y la estabilidad de cada uno de los paquetes del proyecto.

1.4 Frameworks de desarrollo

El concepto de framework (marco de trabajo) tiene muchas acepciones, pero en general, se refiere a una estructura software integrado por componentes personalizables e intercambiables para el desarrollo de una aplicación. Es decir, se puede considerar como una aplicación genérica incompleta y configurable a la que se le puede añadir elementos para desarrollar una aplicación concreta.

Puede decirse que un framework es el esqueleto de una aplicación que debe ser adaptado por el programador para desarrollar una aplicación específica.

Existen diversos frameworks para desarrollo en PHP 5. De entre ellos se seleccionaron los siguientes (cuatro de los más usados), con el propósito de medir sus características y determinar qué tan bien aplican los conceptos de la POO:

- **Yii:** Se destaca por un alto rendimiento basado en componentes para desarrollar aplicaciones web a gran escala. Ofrece casi todas las características necesarias para el desarrollo de aplicaciones web 2.0, tales como MVC, ActiveRecord, servicios web, etc. [18].
- **CodeIgniter**: Posee un diseño compacto para crear aplicaciones web completas. Proporciona un amplio conjunto de bibliotecas para tareas comunes, así como una interfaz simple y estructura lógica para acceder a estas bibliotecas. Permite enfocarse creativamente en el proyecto, reduciendo al mínimo la cantidad de código necesario para una tarea determinada [19].
- **Zend**: se basa en la simplicidad y en las mejores prácticas orientadas a objetos. Se centra en la creación de aplicaciones de web 2.0 seguras, confiables, y consumir APIs disponibles de proveedores líderes como Google, Amazon, Yahoo!, Flickr, entre otros [20].

- **Symfony**: Posee un reducido número de requisitos previos, lo cual hace que sea muy fácil de instalar en cualquier configuración (Linux o Windows). Es compatible con casi cualquier sistema de base de datos. Tiene una baja curva de aprendizaje y permite construir aplicaciones robustas en un contexto empresarial. Incluye herramientas adicionales que ayudan a probar, depurar y documentar el proyecto. Adicionalmente ofrece los beneficios de una activa comunidad de código abierto. Es totalmente gratuito y publicado bajo la licencia MIT [21].

En la tabla 1 se muestra un resumen de las características de los frameworks mencionados [22]:

Tabla 1: Resumen de de las características de los frameworks evaluados

PHP Framework	MVC	Multiple DB's	ORM	DB Objects	Caching	Validation	Ajax	Auth Module	Modules	EDP
CodeIgniter	Si	Si	-	Si	Si	Si	-	-	-	-
Symfony	Si	Si	Si	Si	Si	Si	Si	Si	Si	-
Yii	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si
Zend	Si	Si	Si	Si	Si	Si	Si	Si	Si	-

- MVC: soporta una configuración del Model-View-Controller.
- MultipleDB's: soporta múltiples bases de datos sin tener que cambiar nada.
- ORM: soporta un mapeador de objetos, generalmente una implementación de ActiveRecord.
- DB Objects: incluye otros objetos de base de datos, como un TableGateWay.
- Caching: almacenamiento en caché de objeto.
- Validation: tiene una validación incorporada o el componente de filtrado.
- Ajax: viene con soporte incorporado para Ajax.
- Auth Module: módulo integrado para el manejo de autenticación de usuario.
- Modules: tiene otros módulos, como un canal RSS parser, módulo PDF, etc.
- EDP: incluye programación dirigida por eventos.

2 Metodología

En esta sección se describe la metodología utilizada para la evaluación de los frameworks, con el propósito de contribuir a la calidad del producto software en el desarrollo de aplicaciones orientadas a objetos en PHP.

El objetivo de todo proceso de medición es recopilar indicadores cuantitativos sobre entidades software, siendo una entidad software todo elemento software sobre el que se puede aplicar un proceso de medición y que están caracterizadas por una serie de atributos (tamaño, tiempo, etc.). Para realizar la medición es necesario identificar tanto las entidades como los atributos a medir (Morasca en [23]).

La metodología para el desarrollo de la propuesta tomó como referencia las etapas del proceso de medición planteado por Sommerville [24], y que fuera utilizada en trabajos previos [25]:

1. *Seleccionar las medidas a realizar:* Dado que la medición está orientada a obtener un análisis comparativo global, se realizaron mediciones sobre tamaño, uso de herencia, acoplamiento, abstracción y estabilidad del software.
2. *Selección de los componentes a evaluar*: Se seleccionaron para la evaluación cuatro de los frameworks open source más utilizados entre los desarrolladores PHP:Yii, Zend, Symfony y CodeIgniter [26].
3. *Medir las características de los componentes*: Para la medición se seleccionó la herramienta open source PHPDepend, dado que funciona independientemente del IDE utilizado para la construcción del software y presenta informes completos de una variedad de métricas OO. PHPDepend analiza cada archivo del software medido y genera los siguientes informes:

3.1. Overview Pyramid (OP)

Este análisis muestra una visualización general del proyecto [14], agrupando las métricas implementadas en tres categorías: Herencia (*Inheritance*), Acoplamiento (*Coupling*) y Tamaño y Complejidad (*Size&Complexity*), que se presentan gráficamente en una figura piramidal con la estructura que se muestra en la Figura 1.

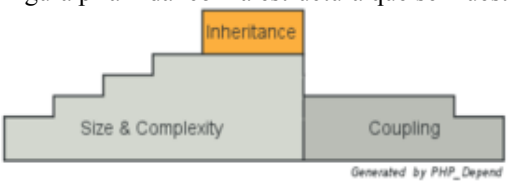


Figura 1. Estructura de componentes de la pirámide

Cada categoría incluye las siguientes métricas [15]:

- Inheritance
  ANDC: Promedio de clases derivadas por clase.
  AHH: Promedio de altura de herencia por clase.
- Coupling:
  CALLS: Cuenta la cantidad de llamadas a métodos diferentes. Si un mismo método es llamado dos veces en la misma clase, es contado solo una vez.
  FANOUT: Cuenta el número de clases colaboradoras que utiliza una clase.
- Size&Complexity:
  NOP: Cuenta la cantidad de paquetes.
  NOC: Cuenta la cantidad de clases.
  NOM: Cuenta la cantidad de métodos.
  LOC: Cuenta la cantidad de líneas de código.
  CYCLO: Suma la complejidad ciclomática de cada método del proyecto.

En cada peldaño de la pirámide se ubica el valor del cociente de normalización aplicado a la métrica de la fila inferior (Ver Figura 2). Por ejemplo, en el caso de NOM (*number of methods*) se divide por NOC (*number of clases*) para adaptar dicha medición al tamaño del proyecto que se está evaluando.

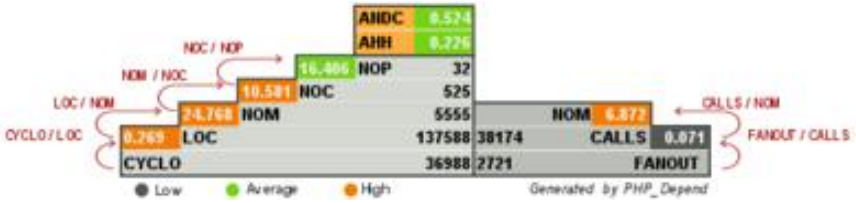


Figura 2. Valores de las métricas y cocientes de normalización en la pirámide

A su vez, los valores tienen un fondo de color que permite visualizar rápidamente las categorías de cumplimiento en cuanto a los valores aceptables, según la escala de intervalos que se muestran en la Tabla 2. Esta escala clasifica como Promedio a aquellas mediciones cuyos resultados son mayores que el indicado como Bajo y menores que el indicado como Alto, para cada medición respectivamente. Los valores umbrales indicados como Bajo y Alto son los que se encuentran por defecto en la herramienta PHPDepend [14].

Los valores clasificados como Bajos se referencian con fondo de color gris oscuro, los clasificados como Altos con fondo color anaranjado y los valores Promedio con fondo de color verde.

Tabla 2: Escala de valores aceptables para las métricas analizadas

Métrica	Bajo	Promedio	Alto
CYCLO/LOC	0,16	0,20	0,24
LOC/NOM	7	10	13
NOM/NOC	4	7	10
NOC/NOP	6	17	26
CALLS/NOM	2,01	2,62	3,2
FANOUT/CALLS	0,56	0,62	0,68
ANDC	0,25	0,41	0,57
AHH	0,09	0,21	0,32

3.2. Abstraction Instability Chart (AI)

La flexibilidad y extensibilidad propias de la POO son factores que hacen a la calidad del software. Estas características dependen en gran medida de un adecuado nivel de acoplamiento entre objetos, lo cual contribuye a la mantenibilidad del software. Por tal motivo, es deseable reducir las dependencias entre las clases. Esto podría lograrse mediante el uso de clases abstractas e interfaces, en lugar de implementación real en la aplicación, lo cual implica algún tipo de contrato. De este modo se flexibiliza el código al permitir a una aplicación implementar sus propias clases para cumplir el contrato. Esta es una característica de la POO que reduce el riesgo de que al modificar una clase/paquete, esto repercuta en el resto del producto.

Esta opción de la herramienta [16] indica la calidad del diseño en términos de extensibilidad, reutilización y mantenibilidad, en función de las dependencias y la abstracción de paquetes, basado en la propuesta de Martin [17]. Este análisis establece

una relación entre la proporción de clases abstractas y la estabilidad de la clase, es decir, trata de definir un valor de estabilidad de acuerdo a la proporción de clases abstractas de cada paquete.

Métricas usadas:

Ca - Acoplamiento aferente: número de paquetes que dependen de clases dentro del paquete analizado. Es un indicador de cómo influye en el resto del proyecto analizado algún cambio en el paquete.

Ce - Acoplamiento eferente: número de paquetes de los cuales dependen clases del paquete analizado. Es un indicador de cuan sensible a cambios en otro lugar del proyecto es un paquete.

I-Instability: Es la proporción entre Ce y el acoplamiento total (Ca+Ce), que se basa en la siguiente formula (Ce/(Ce+Ca)). I=0 indica máxima estabilidad del paquete, es decir no depende de nadie. I=1 indica dependencia total de otros paquetes.

A-Abstractness: Es la proporción entre clases abstractas (AC) y el total de las clases (CC+AC) que se calcula mediante la fórmula (AC/(AC+CC)). A=0 implica que no existen clases abstractas en el paquete mientras que A=1 implica un paquete compuesto solo de clases abstractas.

Es deseable que todos los paquetes se ubiquen sobre la diagonal. La Figura 3 muestra la disposición de los paquetes según las métricas del proyecto analizado.

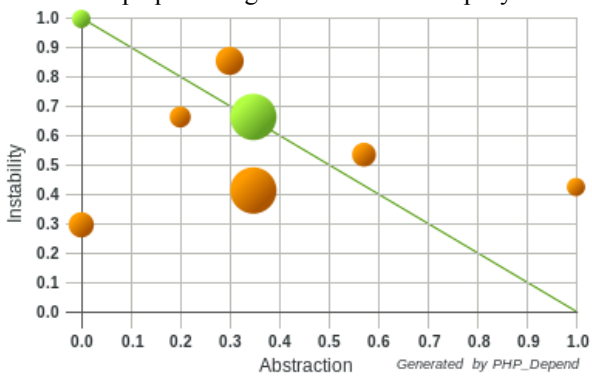
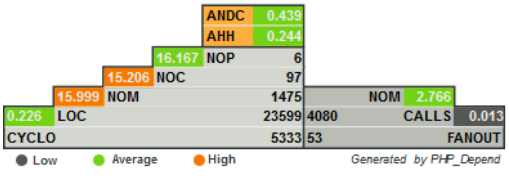
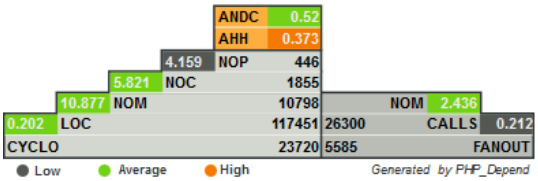
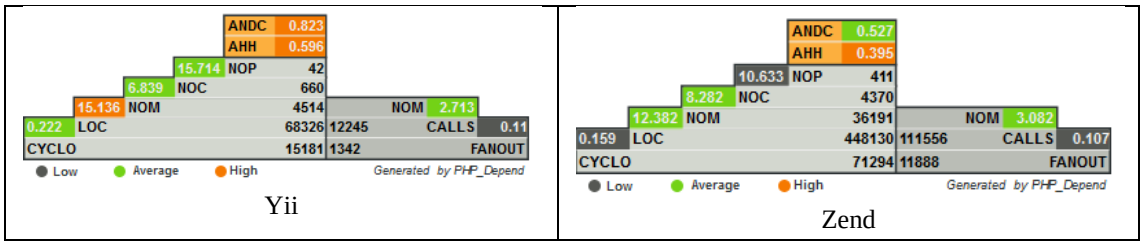


Figura 3. Abstraction Instability Chart (AI)

4. *Identificar las mediciones anómalas:*
   Los gráficos en la Tabla 3 muestran los resultados del análisis de Overview Pyramid (OP) para cada framework evaluado. Los valores se encuentran coloreados según su relación con la escala propuesta: Bajo (gris oscuro), Promedio (verde) y Alto (anaranjado), tal como se indica al pie de la pirámide.

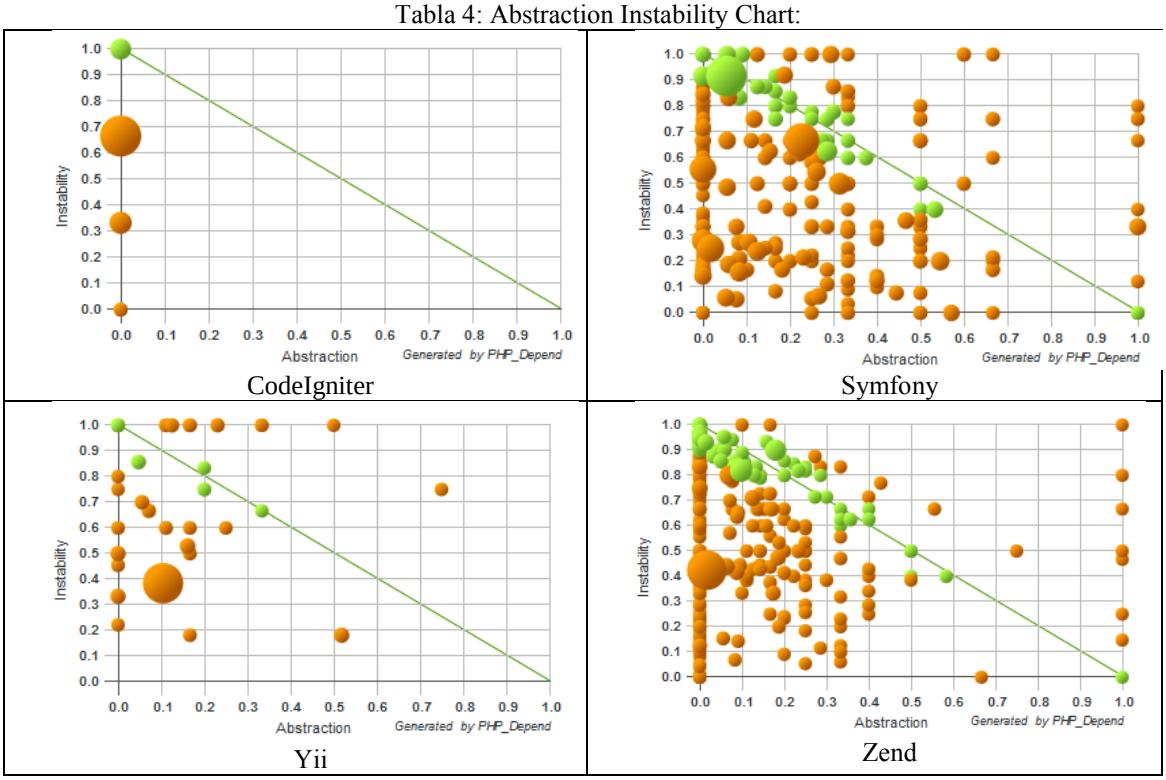
Tabla 3: Análisis OP para cada framework

 CodeIgniter	 Symfony



Respecto al análisis de estabilidad, en la Tabla 4 se muestran los gráficos resultantes para los frameworks analizados:

Tabla 4: Abstraction Instability Chart:



5. *Análisis de los resultados*

Los valores de las métricas obtenidos en el análisis de los distintos frameworks mediante Overview Pyramid (OP) se muestran en la Tabla 5, coloreados según su relación con la escala propuesta (Bajo, Promedio, Alto).

Se destaca que CodeIgniter es el único que posee valores promedio en las dos métricas de herencia. La herencia es una de las características altamente deseadas de la POO, ya que favorecen la reutilización del software. Cabe señalar también que una profundidad adecuada en el árbol de herencia contribuye a un mejor nivel de complejidad, impactando favorablemente en el mantenimiento del software.

Se observa también que todos los frameworks cumplen parcialmente con los valores promedio para el acoplamiento, siendo Zend el más alejado de los mismos.

Para establecer un valor global del grado de calidad de cada framework en función de este análisis se calculó el porcentaje de valores promedio, para lo cual se contaron los comprendidos dentro de los llamados resultados deseados (promedio - color verde) y se dividieron por el total de valores observados.

En la Tabla 5 se puede observar que los frameworks CodeIgniter y Symfony se encuentran mejor posicionados, ya que el 62,5% de los valores observados se encuentran dentro de rangos de valores aceptables (Promedio), mientras que Yii presenta el 50% y Zend sólo el 12.5% dentro de esta categoría.

Tabla 5: valores de las métricas de OP para cada framework

Framework	Tamaño y Complejidad				Herencia		Acoplamiento		% de valores promedios
	CYCLO	LOC	NOM	NOC	AHH	ANDC	CALLS	FANOUT	
Symfony	0,202	10,877	5,821	4,159	0,373	0,52	2,436	0,212	62,5 %
CodeIgniter	0,226	15,999	15,206	16,167	0,244	0,439	2,766	0,013	62,5 %
Zend	0,159	12,382	8,282	10,633	0,395	0,527	3,082	0,107	12,5
Yii	0,222	15,136	6,839	15,714	0,596	0,823	2,713	0,11	50%

Del análisis de estabilidad (AI) se puede concluir que ninguno de los frameworks implementa la totalidad de sus paquetes de manera estable. Sin embargo, cabe señalar que Symfony y Zend presentan la mayor cantidad de paquetes sobre la diagonal.

Considerando la totalidad de las métricas evaluadas, se observa que Symfony reúne las mejores condiciones de cumplimiento de valores aceptables, ya que en el análisis de OP obtuvo un porcentaje del 62,5% y además presenta un resultado favorable en el análisis AI.

3 Conclusiones

Se realizó la medición de atributos de calidad de cuatro frameworks para el desarrollo de aplicaciones PHP, utilizando la herramienta PHPDepend. Del análisis de los resultados surge el siguiente orden de mérito en función del cumplimiento de los umbrales establecidos para las métricas analizadas: Symfony, CodeIgniter, Yii y Zend.

En la actualidad el uso de frameworks está ampliamente difundido entre los programadores, por tanto, utilizar un framework que cumple criterios de calidad, contribuirá a la calidad del producto que se desarrolla.

Como trabajo futuro se propone elaborar una herramienta propia que implemente una mayor cantidad de métricas OO aplicables a desarrollos PHP OO.

4 Referencias

1. Fernández, L.; Lara, P.; Cuadrado-Gallego, J. "Mejora de la calidad en desarrollos orientados a objetos utilizando especificaciones UML para la obtención y precedencia de casos de prueba". Revista de Procesos y Métricas de las Tecnologías de la Información (RPM). ISSN 1698-2029. VOL. 1, Nº 3, Diciembre 2004, 11-20.
2. Hernández Ballesteros, J.F.; Minguet Melián, J. M. "La Medida de la Calidad del Software como Necesidad y Exigencia en Modelos Internacionales (CMMI, ISO 15504, ISO 9001)". www.issi.uned.es/CalidadSoftware/Noticias/PonIng2005.rtf
3. Pressman, R. "Ingeniería de Software. Un enfoque práctico". McGraw-Hill. 2005.
4. Avello, D.G.; Cernuda del Río, A. "Reflexiones y Experiencias sobre la Enseñanza de POO como único Paradigma". JENUI 2003. Cádiz, España.
5. Categories of Programming Languages. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
6. Khaled El Emam, "A Primer on OO Measurement". 1530-1435/05 IEEE. Proceeding of the Seventh International Software Metrics Symposium (ETRICS'01).
7. TIOBE Programming Community Index for July 2012. July Headline: Objective-C overtakes C++. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
8. Manual de PHP. http://www.php.net/manual/es/oop5.intro.php
9. Fernández, A. Comparativa proyectos PHP para e-commerce. Revista Todo Linux. Nº. 112, 2010 , págs. 26-30.
10. PHPMD - PHP Mess Detector. http://phpmd.org/
11. CurrentRulesets. Lista de conjuntos de reglas y normas contenidas en cada grupo de reglas. http://phpmd.org/rules/index.html
12. Sitio oficial de Sonar. http://www.sonarsource.org/
13. What is PHP_Depend? http://pdepend.org/documentation/what-is-php-depend.html
14. Overview Pyramid. http://pdepend.org/documentation/handbook/reports/overview-pyramid.html
15. Lanza, M.; Marinescu, R. "Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems". Object-Oriented Metrics in Practice. Springer-Verlag Berlin Heidelberg; ISBN 978-3-540-24429-5. 2006.
16. Abstraction Instability Chart. http://pdepend.org/documentation/handbook/reports/abstraction-instability-chart.html
17. Martin, R.C. "OO Design Quality Metrics - An Analysis of Dependencies". http://www.objectmentor.com/resources/articles/oodmetrc.pdf. 1994
18. Sitio oficial framework Yii. http://www.yiiframework.com/
19. Sitio oficial framework CodeIgniter. http://www.codeigniter.com/
20. Sitio oficial framework Zend. http://framework.zend.com/
21. Sitio oficial framework Symfony. http://www.symfony-project.com/
22. PHP Frameworks. http://www.phpframeworks.com/
23. Piattini M., García O., Caballero I. "Calidad de los sistemas informáticos". Ed. RA-MA. MADRID, España. 2007.
24. Sommerville, I. "Ingeniería del Software". Séptima Edición. Madrid: Pearson Educación, S.A., 2005. 84-7829-074-5
25. Greiner C.; Demchum, D.; Estayno, M.; Dapozo G. "Una propuesta de solución para automatizar la medición de aplicaciones orientadas a objeto". Anales del XVI Congreso Argentino de Ciencias de la Computación 2010 (CACIC2010). ISBN 978-950-9474-49-9. Pag. 654-663. Facultad de Informática, Ciencias de la Comunicación y Técnicas Especiales de la Universidad de Morón. 2010.
26. Top 10 Ranking PHP Frameworks. http://www.phpframeworks.com/top-10-php-frameworks/