

MApache, estilo de programación para el desarrollo rápido de aplicaciones ingenieriles orientadas a la web

Carlos Armando López S.

Recibido el 15 de febrero de 2011. Aprobado el 10 de mayo de 2011

Resumen

Un estilo es la forma o modo como se hacen las cosas, obteniéndose como resultado un objeto (producto) con características particulares. Al tener un procedimiento, un grupo de lenguajes y una organización típicas en la creación de software, se pueden generar costumbres que faciliten el desarrollo de aplicaciones con un carácter propio.

MApache es un estilo de programación desarrollado por el autor con ayuda de la comunidad de software libre de UNIMINUTO, Arca-csl, que busca cambiar el enfoque acerca del uso de APACHE server. La idea es ver a este software, no solamente como el poderoso servidor que es, sino como una máquina virtual tal como sucede con Java VM o CLI de .NET.

Para definir el estilo de programación MApache se han establecido varias políticas de desarrollo así como algunas herramientas de software (Paquetes de clases utilitarias) con las cuales se busca velocidad y facilidad a la hora de desarrollar aplicaciones orientadas a la web. Adicionalmente, se busca con estas políticas y herramientas promover el desarrollo de aplicaciones especializadas en ingeniería.

Este artículo busca explicar el estilo de programación MApache y su utilización en proyectos de software orientados a la web. También presenta la posibilidad de que un estilo de programación genere, debido a la experiencia y a la implementación de patrones, frameworks que permitan estandarizar el desarrollo de aplicaciones.

Palabras clave

PHP, Frameworks, Programación orientada a la web, software especializado

Abstract

A style is the way things are done, resulting in an object (product) with particular characteristics. By having a procedure, a group of languages and a typical organization in creating software, we can create habits that facilitate application development with its own character. MApache is a programming style developed by the author with help from the UNIMINUTO free software community, Arca-CSL, which seeks to shift the focus on the use of APACHE server. The idea is to see this software, not only as the powerful server, but as a virtual machine as with Java VM or CLI. NET.

To define the MApache style programming, have set up several development policies as well as some software tools (package utility classes) seeking speed and ease in developing web-oriented applications. Additionally, for these policies and tools to promote the development of specialized applications in engineering.

This article explain the programming style and its use in Mapache web-oriented software projects. It also presents the possibility of generating a programming style, due to the experience and implementation of standards, frameworks that enable to standardize application development.

Keywords

PHP, Frameworks, Web Programming Oriented, specialized software

I. Introducción

Los estilos de trabajo en la industria, permiten que el resultado final se distinga de otros fabricantes, por ejemplo, en la industria automotriz, se pueden identificar elementos como el carácter y la personalidad de un vehículo, dependiendo de la marca (es decir, de la empresa fabricante) la cual tiene procedimientos propios e incluso secretos para su fabricación.

Al tener un procedimiento, un grupo de lenguajes y una organización típicas en la creación de software se pueden generar costumbres que faciliten el desarrollo de aplicaciones. Esto permite también la creación de técnicas que ayudan al fortalecimiento del conocimiento de la programación en general.

En el desarrollo de software que surge al interior de la comunidad de software libre, Arca-csl, se han implementado varios estándares y políticas que en conjunto dan como resultado un estilo de programación que se ha denominado MApache¹.

MApache ha sido creado pensando en la programación orientada a la web y en la utilización de lenguajes de programación como PHP y Javascript, y se espera que en un futuro, a través de la comunidad universitaria, sea utilizado a través de patrones que permitan generar un framework que acelere y estandarice el desarrollo de las aplicaciones.

II. Estilo de programación

Al programar, se tienen costumbres y formas de solucionar los problemas que se forjan a través de la experiencia del programador. Cuando una costumbre o forma de solucionar un problema se populariza, se tiende a producir estándares que luego se convierten en lo que se conoce dentro del mundo de la programación como patrones².

1 MApache busca cambiar el enfoque tradicional del servidor Apache, para verlo también como una máquina virtual al igual que Java VM, aprovechando tecnologías como Prism de Mozilla, La M de MApache simboliza ese hecho, M de Máquina.

2 Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón se debe haber comprobado la solución de problemas similares en ocasiones anteriores con él, y debe ser reutilizable en distintas circunstancias.

Sin embargo, la implementación de patrones en el desarrollo de aplicaciones, no es una receta ni un grupo de reglas estrictas, sino más bien un grupo de políticas que buscan solucionar los problemas que surgen en el desarrollo de aplicaciones (Freeman *et al.*, 2004).

Así, un estilo de programación debe incluir no solo la estrategia para afrontar el desarrollo de software sino la forma como los patrones ya conocidos se implementan, promoviendo la creación de herramientas que faciliten el desarrollo de aplicaciones (Frameworks³) y permitan cumplir con los patrones que se requieran manteniendo las características deseadas de las mismas.

Por otro lado, los frameworks tienen un comportamiento dependiente del lenguaje con el que se han desarrollado, afectando así las aplicaciones que se crean con ellos. En otras palabras, los lenguajes de programación, por su naturaleza, tienden a afectar la forma como se hacen los frameworks y estos a su vez afectan a las aplicaciones, un ejemplo de ello se puede ver en el proyecto Kumbia (KumbiaPHP, s.f.).

Por lo general, un estilo de programación afecta las funcionalidades (requerimientos funcionales) y la calidad (Requerimientos no funcionales) de la aplicación, con especial énfasis en esta última, dándole un carácter único.

Para mantener la calidad y el carácter único de una aplicación, el estilo de programación debe procurar:

- Mantener costumbres "sanas" de programación (Buenas prácticas)
- Asimilar las técnicas de programación que se acomoden a las necesidades
- Promover la creación de técnicas propias
- Promover el análisis de lenguajes que se acomoden a las necesidades
- Promover la utilización de patrones de desarrollo
- Disciplina de grupo
- Promover la experimentación de nuevas ideas

3 Un framework puede considerarse como una herramienta de software que facilita la creación de software

Un estilo de programación afecta principalmente:

- La ergonomía de la GUI (Graphic User Intefrace)
- Tiempos de respuesta
- La estructuración interna de la codificación

Un ejemplo de esto se puede apreciar en las aplicaciones creadas por grandes fabricantes como Microsoft, donde todos sus productos de software tienen tiempos de respuesta similares, un aspecto similar, se relacionan fácilmente entre sí y por lo general evolucionan en conjunto.

Esto ocurre porque las personas involucradas en la creación y mantenimiento de las aplicaciones en Microsoft tienen políticas, reglas y herramientas de software (frameworks) que les permiten estandarizar su trabajo.

III. Estilo de programación MApache

MApache es un estilo de programación, en el que se han establecido varias políticas de desarrollo así como algunas herramientas de software (Paquetes de clases utilitarias) con las cuales se busca velocidad y facilidad a la hora de desarrollar aplicaciones orientadas a la web. Adicionalmente, se busca con estas políticas y herramientas promover el desarrollo de aplicaciones especializadas en ingeniería.

Las clases MApache se han organizado en dos paquetes:

Weblinx: El cual permite la creación de interfaces gráficas (GUIs) siguiendo el paradigma de programación orientado a objetos, de forma similar a como se crean interfaces gráficas en lenguaje JAVA mediante librerías especializadas como Swing (Loy *et al.*, 2002) y AWT.

Englinx: El cual permite el desarrollo de cálculos especializados en ingeniería, incorporando métodos numéricos.

MApache aprovecha tecnologías como Prims⁴ de Mozilla (Prism, 2010) y AJAX⁵, para que Apache se pueda asimilar por parte del programador como una máquina virtual en la cual se pueden ejecutar aplicaciones de cualquier tipo sin importar si requieren acceso o no, a Internet.

4 Prism es un proyecto de la organización Mozilla que busca que las aplicaciones web tengan un comportamiento similar a las aplicaciones stand alone, es decir, sean similares a las aplicaciones locales en un PC.

5 Ajax, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor.

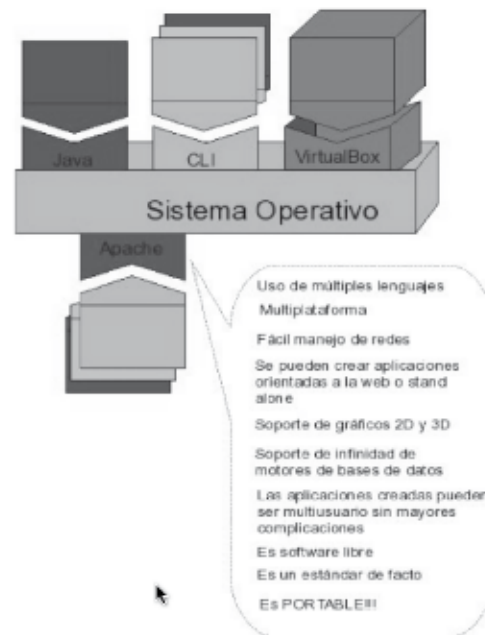


Figura 1. Esquema representativo del uso de Apache como máquina virtual. Fuente: el autor. 2007

Dentro de las políticas que se promueven en el desarrollo de aplicaciones usando MApache están:

1. Una aplicación MApache debe seguir como mínimo, el patrón Modelo Vista Controlador (MVC) (Freeman *et al.*, 2004) basado en la organización de archivos.
2. Una aplicación MApache debe estar hecha con lenguajes de programación orientados a la web, en especial se recomienda el uso de PHP.
3. Una aplicación MApache procura tener una apariencia stand alone⁶.
4. Una aplicación MApache puede cambiar la apariencia de su GUI de acuerdo a temas y estilos definidos en CSS⁷.
5. Una aplicación MApache debe ser construida con las clases Weblinx y Englinx.
6. Una aplicación MApache debe ser software libre.
7. Una aplicación MApache debe ser traducible a varios idiomas.
8. Una aplicación MApache, de acuerdo a los requerimientos, debe seguir patrones de diseño que sean convenientes.
9. Si se descubre en el desarrollo de una aplicación MApache la necesidad de crear una herramienta de software nueva (una utilidad nueva), la cual puede ser necesaria en otros desarrollos, se deberá crear e incluir dentro de los paquetes de clases MApache.

6 Hace referencia a una aplicación que no requiere acceso a Internet, son aplicaciones que no requieren de navegador web para funcionar.

7 Los CSS son archivos que definen la presentación de elementos xhtml o de otros lenguajes de etiquetas de presentación.

10. Toda nueva utilidad MApache que requiera el uso de lenguajes diferentes a PHP deberá ser creada con un front-end⁹ hecho en PHP.

Un front-end en programación hace referencia a la utilización de código en un lenguaje para comunicar solicitudes a una aplicación que está hecha en otro lenguaje.

Debido a su facilidad de uso, la creación de aplicaciones puede ser más agradable que cuando se desarrollan con otros lenguajes.

Sus principales ventajas son las siguientes:

- Es multiparadigma
- Fácil de aprender
- No depende de entornos de desarrollo
- Promueve el software libre por ser interpretado
- Es multiplataforma gracias a Apache
- Tiene un gran soporte para bases de datos
- Tiene soporte para la creación de gráficos
- Tiene soporte para comunicaciones

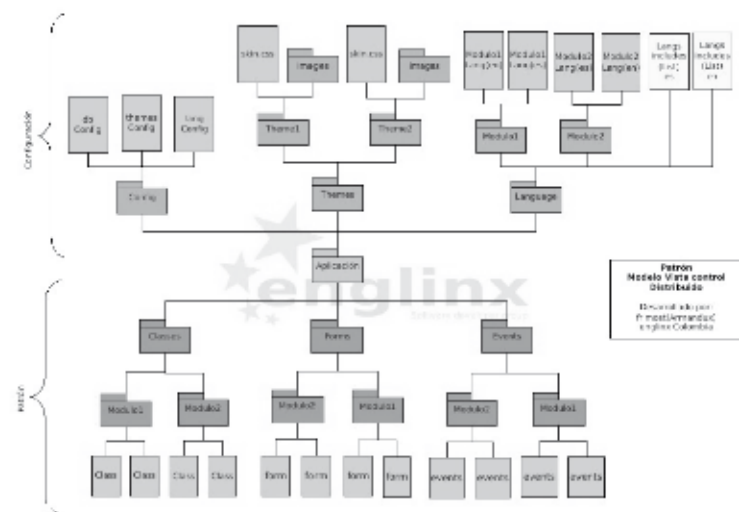


Figura 2. Estructura de archivos de una aplicación MApache. Fuente: el autor. 2007

Esta estructura de aplicación permite hacer énfasis en el uso del patrón Modelo Vista Controlador (MVC)⁹ y separa el software necesario para la configuración de las funcionalidades.

Dentro del área de configuración (ver figura 2) se encuentran paquetes de clases y scripts que permitan configurar rutas de conexión a bases de datos, a temas de presentación, idiomas, entre otros.

Dentro del área del patrón MVC (ver figura 2) se estructura toda la codificación del modelo y presentación de la aplicación.

IV. ¿Por qué usar PHP?

Existen muchos lenguajes soportados por Apache server, sin embargo, el lenguaje que se ha decidido utilizar para el desarrollo de MApache es PHP.

PHP (Sklar, 2004) es un lenguaje de programación muy versátil y de gran capacidad de adaptación, es fácil de aprender y de "naturaleza relajada".

9 El MVC es un patrón de diseño que busca separar la parte lógica de la presentación en una aplicación. Es muy utilizado en la programación web, ya que optimiza la implementación de la arquitectura Cliente-Servidor

V. Grupo de clases (Weblinx y Englinx)

Como se mencionó anteriormente, al programar usando el estilo de programación MApache se requiere del uso de los paquetes de clases Weblinx y Englinx (López, 2007)

Weblinx contiene clases que permite la creación de elementos html como áreas de texto, botones, formularios, etc. También incluye clases que permiten la validación de datos ingresados en formularios y la presentación de gráficos en 2D.

Por su parte, Englinx, permite la creación de objetos que realizan cálculos ingenieriles como: La solución de ecuaciones lineales, el cálculo de áreas, centros de gravedad, conversión de unidades, etc.

¿Cómo funcionan?

Aprovechando que el servidor Apache puede soportar diversos lenguajes de programación, se pueden crear aplicaciones multilenguaje.

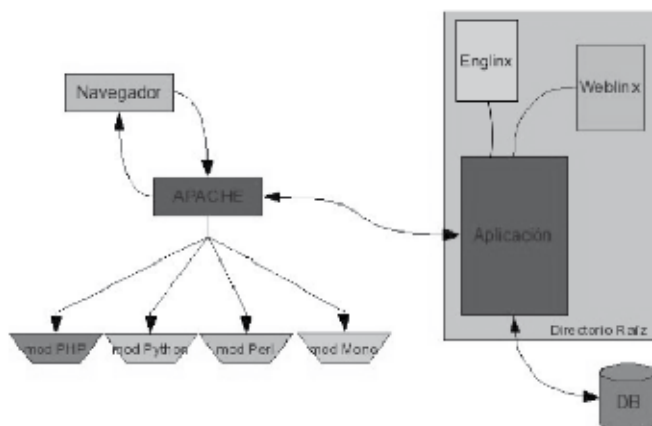


Figura 3. Esquema acerca del uso de los lenguajes soportados por Apache y la aplicación de las clases MApache. Fuente: El autor. 2007

```
<?php
require '../MAPache/Weblinx.php';
//Creación de un formulario simple:-----
-----//
$this->easyForm=new EasyForm("entitiesForm", "en
titiesForm","get","../Events/Embargoes/Events-
FrmEntities.php","Registro de nueva Persona Le-
gal:","","","");
$this->easyForm->addInput(1,"Cedula o NIT:","id",
"id","","","","","form","","",array());
$this->easyForm->addInput(1, "Dirección:","direcc
ion","direccion","","","","","form");$ t h i s -
>easyForm->addInput(1, "Telefono:","telefono","te
lefono","","","","","form");
$this->easyForm->addInput(1, "Ciudad:","ciudad","
ciudad","","","","","form");
$this->easyForm->addInput(1, "Email:","email","em
ail","","","","","form");
$this->easyForm->addButton(12,"","RegistrarE","Re
gistrarE","registrarE","registrar",
"registrar","button","","button","","Registrar
persona legal","","","0");
//Presentación del formulario:-----
-----//
$this->easyForm->show(6,1);
?>
```

En el ejemplo presentado en la figura 4, se ha creado un formulario con un objeto de la clase EasyForm de Weblinx de MAPache, que permite la captura de datos y tiene un aspecto como el siguiente (usando cualquier navegador web):

Registro de nueva Persona Legal:

Cédula o NIT:	<input type="text"/>
Dirección:	<input type="text"/>
Teléfono	<input type="text"/>
Ciudad	<input type="text"/>
E-mail	<input type="text"/>

Es común encontrar en los formularios de aplicaciones web mezclas entre lenguajes de programación y presentación. Por ejemplo, se quiere validar los datos que se ingresan en este formulario hecho en xhtml, es necesario utilizar el lenguaje de programación Javascript (Flanagan, 2006) para ello. Y si alguno de los

Al usar EasyForm de MApache, se impone un único lenguaje (PHP) manteniendo un orden en la codificación. Si se quiere hacer validaciones de los datos que se ingresan al formulario, se puede utilizar la clase Validator de MApache la cual permite múltiples reglas de validación:

Figura 6. Creación de un formulario web con validaciones usando las clases weblinx de MApache. Fuente el autor. 2009

Se espera que el estilo de programación MAPache, en el futuro, tenga su propio framework gracias a la evolución de los paquetes de clases y a la incorporación de patrones en las políticas de desarrollo.

Las clases MApache se actualizan periódicamente incluyendo nuevas funcionalidades y corrigiendo bugs,

gracias a los proyectos de programación promovidos por la comunidad de software libre Arca-csl.

VI. Ventajas de usar MApache (López, 2007)

1. Limpieza en el código

Ya que con el uso del paquete de clases Weblinx es posible eliminar el uso de código ajeno a PHP, en especial en la creación de la interfaz gráfica.

2. Uso de la programación orientada a objetos en PHP

Al usar el paradigma de programación orientado a objetos se reduce aún más el desorden en el código. Por otro lado, permite la reutilización del mismo en otras aplicaciones.

3. Flexibilidad

Gracias al uso del Patrón Modelo Vista Controlador, entre otros patrones, se permite que las aplicaciones sean fácilmente mejoradas.

4. Reducción en el tiempo de codificación.

Como se procura un orden estricto en la conformación de los archivos de la aplicación mediante el uso de patrones, es posible la reutilización de código gracias a la programación orientada a objetos y a la eliminación de las complejas relaciones entre otros lenguajes (como javascript y html) con PHP. Así, mediante las clases Weblinx, es posible reducir en gran medida los tiempos de codificación.

El estilo de programación MApache es utilizado en el proyecto *engineer+* (López, 2010) de Arca-csl.

engineer+ busca la creación de una herramienta (Servicio web) modular especializada en cálculos ingenieriles. Cada componente de *engineer+* se desarrolla de forma independiente siguiendo estrictamente las políticas de MApache y utilizando los paquetes de clases Weblinx y Englinx.

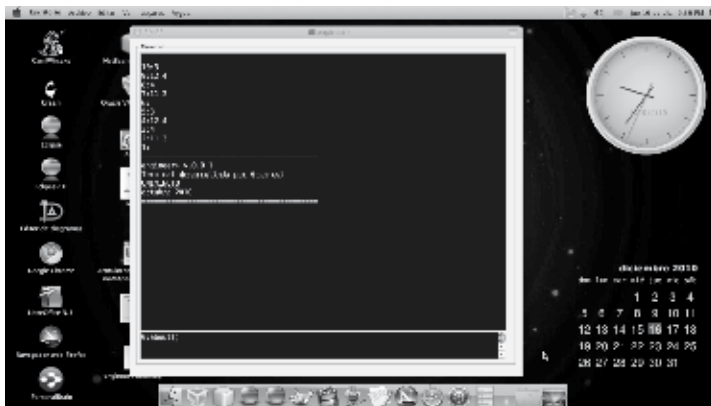


Figura 7. Terminal de *engineer+* creada siguiendo los lineamientos de MApache. Fuente: el autor. 2009

En este proyecto se ha dado paso al desarrollo de clases utilitarias para el paquete Englinx, en donde se realizan cálculos ingenieriles. Adicionalmente, se ha creado la clase Terminal en el paquete Weblinx donde se utiliza el concepto de pila (Stack) para la captura de datos en Javascript.

El uso de las clases Englinx es similar al de Weblinx:

```
<?php
require '../MApache/Englinx.php';
//Calculo de area de un poligono:-----
-----//
$geometrics=new Geometrics();
$A= $geometrics->calcAreaPo-
lygon(3.2,4.0,5.0,3.4,20.0,3.2);
.
.
.
?>
```

Figura 8. Uso de la clase *Geometrics* de MApache para el cálculo del área de un polígono. Fuente el autor. 2009

Se espera que el proyecto *engineer+* promueva la creación de clases del paquete Englinx y que se puedan corregir bugs de las clases ya creadas.

VIII. Conclusiones

- La creación de un estilo propio de programación permite generar un framework que facilite el trabajo a través de la experiencia, la retroalimentación y la propuesta de soluciones.
- MApache demuestra que es posible mantener un código limpio y de fácil revisión a pesar de requerir diversas tecnologías y lenguajes de programación.
- MApache ha permitido comprobar que el uso de patrones de diseño debe ser flexible siendo opcional su utilización, dependiendo de los problemas que se desean cubrir, sin embargo, en la programación orientada a la web, resulta conveniente el uso del patrón Modelo Vista Controlador, ya que mantiene el orden y la coherencia frente al uso de la arquitectura Cliente-Servidor.
- La aplicación continua y realimentación al autor, de las políticas y herramientas MApache, permitirá que se corrijan errores y se incorporen nuevas herramientas en su framework.

IX. Referencias

- [1] Flanagan D. (2011), *JavaScript: The definitive guide*, 6th Edition, Sebastopol, O'Reilly Media Inc.

- [2] Freeman, E.; Freeman, E.; Bates, B & Sierra, K. (2004), *Head first Design Patterns*, Sebastopol, O'Reilly Media Inc.
- [3] Kumbia PHP (s.f.), "KumbiaPHP Framework", disponible en: <http://wiki.kumbiaphp.com/Principal>, recuperado: 16 de diciembre de 2010.
- [5] López, C., (2007), "MApache", disponible en: <http://sites.google.com/site/arcacslsite/documentos/MApache.pdf?attredirects=0&d=1>, recuperado: 16 de mayo de 2011.
- [6] López, C (2010), "Engineer+", disponible en: <https://sites.google.com/site/englinx/>, recuperado: 16 de mayo de 2011.
- [7] Loy, M. et al. (2002), *Java Swing*, second edition, Sebastopol, O'Reilly Media Inc.
- [8] Prism. (2010), Proyecto de laboratorio Prism de Mozilla, disponible en: <http://prism.mozillalabs.com/>, recuperado: 16 de mayo de 2011.
- [9] Sklar, D. (2004), *Learning PHP5 A pain-free introduction to building interactive web sites*, Sebastopol, O'Reilly Media Inc.

Carlos Armando López S. Ingeniero Civil de la Universidad De La Salle (1998), Tecnólogo en Informática de la Corporación Universitaria Minuto de Dios (UNIMINUTO) (2007), estudiante de Maestría en Ingeniería de Sistemas y Computación de la Universidad de los Andes. Miembro Fundador de la comunidad de software libre Arca-csl de UNIMINUTO y líder del proyecto engineer+ del grupo de investigación Investigaciones IT de la Facultad de Ingeniería de UNIMINUTO. Actualmente se desempeña como docente en el área de programación en el programa de Tecnología en Informática de UNIMINUTO.