PLAN DE CALIDAD PARA DESARROLLO DE SOFTWARE

Diciembre de 2021

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.

Revisión histórica

Fecha	Versión	Descripción	Autor
13-09-21	1.0	Versión inicial	Viviana García Mauris Antonio Ávila

Tabla de contenido

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.

Tabla de figuras

Tablas

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.

1. Introducción

Canal Capital planteo en su Plan Estratégico de Tecnologías de la Información PETI 2017-2020 en sus rupturas estratégicas la capacidad para implementar y hacer uso de recursos de software a través del desarrollo en casa, debido a las dificultades técnicas, financieras y procedimentales que presenta la adquisición de software propietario.

Diseñada la estrategia para implementar componentes de software desarrollados en la entidad, el área de sistemas definió cuales serían las mejores prácticas y recursos de arquitectura que permitieran brindar un ciclo de vida optimo a los productos desarrollados con base en la premisa de calidad, seguridad, fiabilidad y disponibilidad.

La adecuada definición de un proceso de software aumenta la posibilidad de que un proyecto obtenga un producto de calidad aceptable. En la presente entrega se mostrarán los planes de calidad como instrumento para documentar el proceso de software en un proyecto determinado. Los planes de la calidad pueden tener características diferentes en función de la metodología elegida.

2. Descripción general del proyecto

2.1 Propósito

El propósito de este plan de calidad para desarrollo de software es definir las actividades de desarrollo, los términos y la implementación de un sistema de registro integrados por módulos para **Canal Capital** de forma virtual**.**

2.2 Alcance

El plan de calidad para desarrollo de software describe la metodología y arquitectura a implementar en **Canal Capital** para el desarrollo de software y cómo se adapta de acuerdo a los pormenores del proyecto; los planes de calidad como instrumento para formalizarla adaptación de los procesos y la capacidad de implementación y despliegue en la entidad.

2.3 Objetivo General

Desarrollar software a la medida con base en la transformación de los procesos vigentes a través de metodologías de desarrollo ágil que involucran la participación del usuario, obteniendo componentes de software de rápida implementación y uso.

2.4 Objetivos Específicos

1. Optimizar los procesos y procedimientos susceptibles a ser automatizados a través de software.
2. Implementar una metodología de desarrollo ágil (scrum) que permita diseñar componentes de software de calidad basados en el usuario final.
3. Construir software orientado a componentes que permita la integración y uso unificado de la información (ERP).
4. Hacer uso de herramientas de software de libre distribución para la construcción de los módulos componentes.
5. Proveer un ciclo de vida optimo a los sistemas desarrollados, a partir de la arquitectura cliente servidor, la orientación a multiplataforma y multidispositivo (responsive site).

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.

2.5 Productos de trabajo del proyecto

Los siguientes productos de trabajo se producirán durante el proyecto:

- Manual de usuario.
- Manual de instalación.
- Código fuente (elementos de implementación) de los componentes de software.
- Ambiente de desarrollo y prueba
- Ambiente de producción.
- Documento de arquitectura de software.
- Guía de desarrollo.
- Especificación complementaria (videos, capacitaciones e infogramas).

3. Arquitectura y componentes de software

3.1 Modelo vista controlador (MVC)

Es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

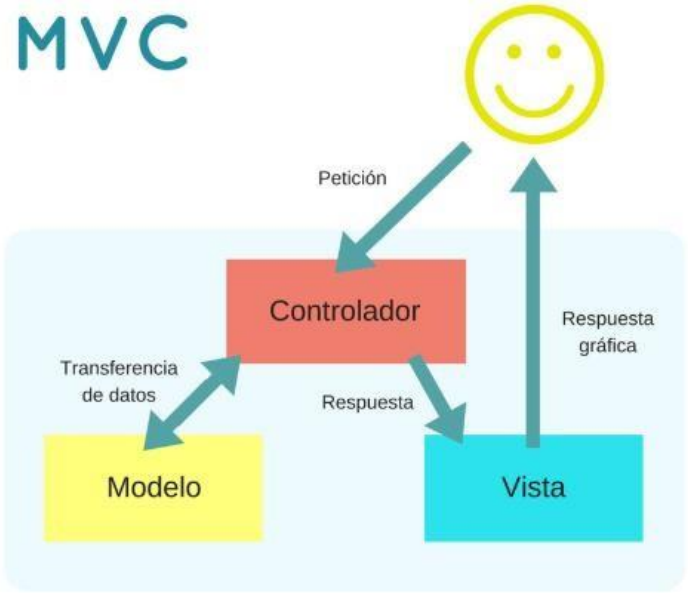
Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo.

El *Modelo* que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.

La *Vista*, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste.

El *Controlador*, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.



4.1 Figura MVC

3.2 Flujo MVC

El usuario interactúa con la interfaz de usuario de alguna forma.

El controlador recibe la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.

El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario. Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.

El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario.

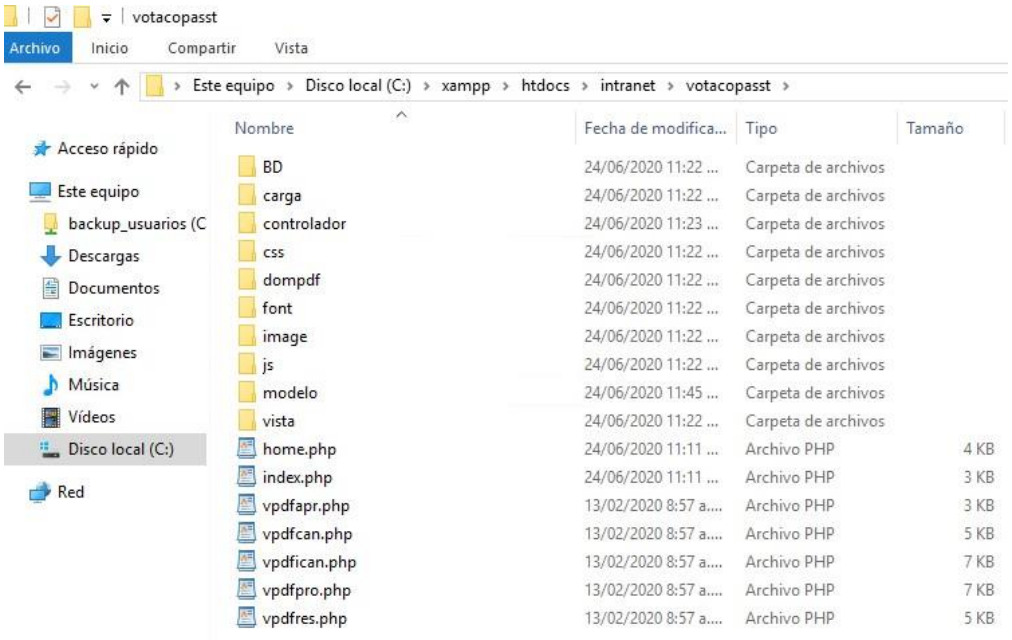
La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo.

La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

Se utiliza todo de forma modular para facilitar la adición de nuevos procesos, facilita las pruebas y permite encontrar y manejar los errores de una mejora manera.

Dentro del canal se organiza la carpeta de la siguiente manera:

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.



4.2 Figura Carpeta Flujo MVC

Dentro de la ruta c:\xampp\htdocs\intranet\ se coloca el nombre de la carpeta del proceso o producto a desarrollar, allí encontraremos las siguientes carpetas:

- **Modelo:** Se encuentran todas las clases que permiten la conexión a base dedatos y consultas.
- **Controlador:** Se encuentran todas las variables, funciones y lógica de cadauno de los procesos necesarios.
- **Vista:** En esta carpeta se almacenan los llamados al controlador y a las funciones que muestran la información solicitada por el usuario.
- **BD:** Script de la base de datos y procedimientos almacenados para el funcionamiento de la base de datos en MySQL.
- **Css:** Páginas de estilo para el diseño del sitio, incluyendo el framework de diseño Bootstrap
- **Js:** Archivos JavaScript para el funcionamiento de validaciones y diferentes funcionalidades del lado del cliente.
- **Font:** Se almacenan los tipos de letras necesarios para el desarrollo.
- **Image:** Se encuentran todas las imágenes que se utilizan en el software incluyendo iconos, banner, logos, entre otros.
- **Otras carpetas:** Se crean de acuerdo al proyecto o necesidades del software.

3.3 Lenguajes de programación a utilizar PHP

Acrónimo recursivo en inglés de PHP: Hypertext Preprocessor (preprocesador de hipertexto), es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en un documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera el HTML resultante.

CSS

CSS (siglas en inglés de Cascading Style Sheets), en español "Hojas de estilo en cascada", es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritasen HTML.

JS

JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

Bootstrap

Es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales. A diferencia de muchos frameworks web, solo se ocupa del desarrollo front-end.

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.

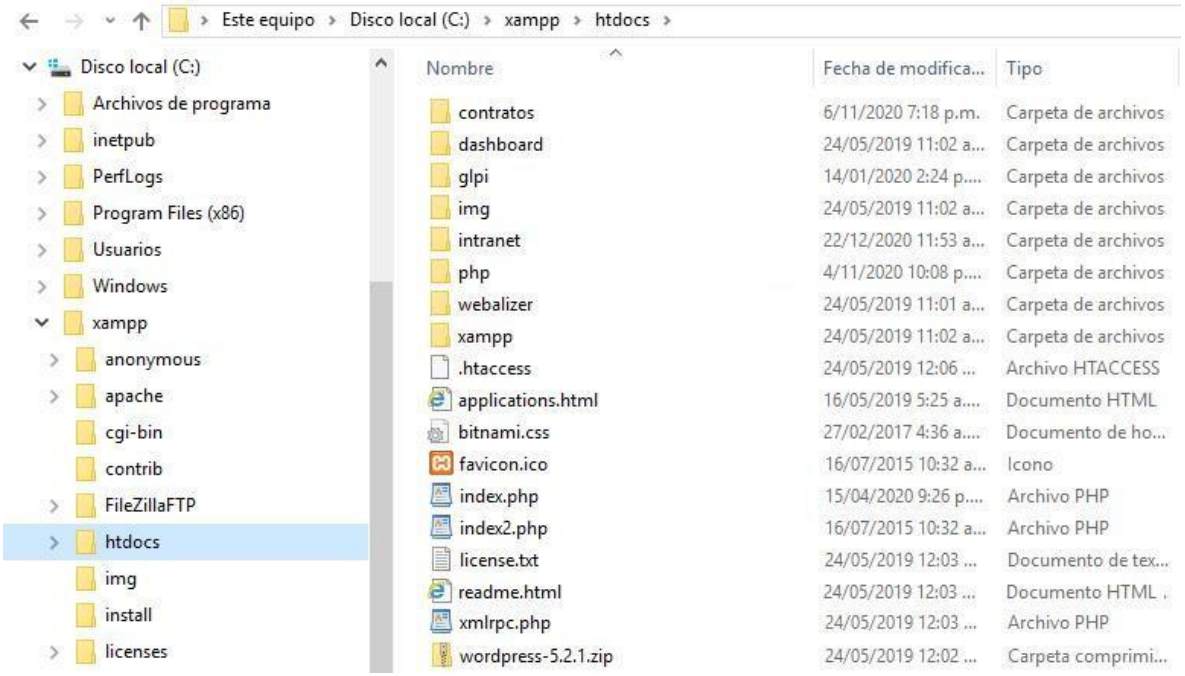
4. Requerimientos de Desarrollo para Canal Capital

En los desarrollos actuales para Canal Capital se está desarrollado en lenguaje PHP con MYSQL, bajo una plataforma CMS WORDPRESS (Sistema administrador de contenidos, sobre la cual se encuentra la intranet desde este se realiza el llamado y acceso al software) y arquitectura MVC (Modelo, Vista, Controlador), con base dedatos diseñada a la medida.

4.1 Intranet

La intranet se encuentra en un servidor virtualizado el cual tiene copias de seguridaden caso de siniestro; además se tiene instalado el servidor de internet Apache, MySQL, PHPMYADMIN (Por medio del instalador XAMPP).

Con la siguiente estructura:



5.1 Figura Carpeta Intranet

4.2 Ayudas del software

Se realiza por medio de video tutoriales en cada uno de los formularios o vistas que se encuentran desarrollados, además que el área de sistema da soporte a los módulos desarrollados.

5. Plan de calidad por fases

El desarrollo del sistema de registro se llevará a cabo utilizando un enfoque por fases en el que se producen múltiples módulos. A continuación, se muestra un resumen del cronograma relativo en la siguiente tabla:

Orden de construcción de los módulos	Nombre modulo	Fin
1	Contratos	6 meses
2	Estructura ERP	2 meses
3	Configuración	3 meses
4	Soporte	1 mes
5	Pasantes	2 semanas
6	Denuncia	2 semanas
7	Contratos migración	3 meses
8	Financiera	5 meses
9	Radicación	1 mes
10	Documentación	1 mes

Tabla 6.1 Modulos ERP

Los proyectos involucran un cierto grado de incertidumbre que conlleva la imposibilidad de prever todos los escenarios posibles al definir el proceso de software, por lo que será necesario revisar la aplicación y adecuación del plan de calidad a la realidad del proyecto para mantenerlo actualizado a medida que se va avanzando en su ejecución.

Todos los productos y módulos de trabajo deben pasar por el proceso de revisión apropiado. La revisión es necesaria para garantizar cada producto de trabajo de calidad, utilizando las pautas y listas de verificación. Además, los defectos se registrarán y rastrearán, y se recopilarán métricas de defectos como se describe en el sitio web del proceso de software en el módulo desoporte.

Dado que el plan de la calidad es un instrumento de apoyo de un proyecto para la definición de su forma de trabajo, estos deben adecuarse al tipo de proceso o metodología que utiliza cada equipo para el desarrollo de software.

FASE	DESCRIPCIÓN	HITO
Fase de comienzo	La Fase Inicial desarrollará los requisitos del producto y establecerá el caso comercial para los diferentes sistemas de registros. Se desarrollarán los principales casos de uso, así como el Plan de Desarrollo de Software de alto nivel.	**Revisión de casos de negocio** al final de la fase marca la decisión de continuar o no para el proyecto.
Fase de elaboración	La Fase de Elaboración analizará losrequisitos y desarrollará el prototipo arquitectónico. Al finalizar la fase de elaboración, todos los casos de uso seleccionados habrán completado el análisis y el diseño. Además, se habrán analizado y diseñado los casos de uso de alto riesgo para la versión 2.0. El prototipo arquitectónico probará la viabilidad y el rendimiento de la arquitectura que se requiere para la versión 1.0.	**Prototipo arquitectónico** marca el final de la fase de elaboración. Este prototipo significa la verificación de los principales componentes arquitectónicos que componen la versión 1.0.
Fase de construcción	Durante la fase de construcción, se analizarán y diseñarán los casos de uso restantes. La versión de la 1.0 se desarrollará y distribuirá para su evaluación. Se completarán las actividades de implementación y prueba para respaldar las versiones1.0 y 2.0.	**Capacidad operativa inicial** (finalización de la versión) marca el final de la fase deconstrucción.
Fase de transición	La versión Beta de la versión 1.0 se distribuirá y evaluará. La fase de transición preparará las versiones 1.0 y 2.0 para su distribución. Proporciona el soporte necesario para garantizar una instalación sin problemas, incluida la formación del usuario.	**2.0 lanzamiento** marca el final de la fase de transición. En este punto, todas las capacidades, tal como se definen en el documento, están instaladas y disponibles para los usuarios.

Tabla 6.2 Fases desarrollo software

5.1 Fases del proyecto e hitos principales

Cada fase se divide en iteraciones de desarrollo en las que se desarrolla un subconjunto del sistema. En general, estas iteraciones:

- Reducir el riesgo técnico
- Proporcionar las primeras versiones de un sistema en funcionamiento
- Permita la máxima flexibilidad en la planificación de funciones para cada versión
- Permita que los cambios de alcance se manejen de manera efectiva dentrode un ciclo de iteración.

La siguiente tabla describe las iteraciones junto con los hitos asociados y los riesgos abordados:

Fase	Iteración	Descripción	Hitos asociados	Riesgos abordados
Fase de comienzo	Preliminar	Define el modelo de negocio, los requisitos del producto, el plan de desarrollo de software y el caso de negocio.	Revisión de casos comerciales	Aclara los requisitos del usuario desde el principio. Desarrolla Planes de Desarrollo de Software y alcance realistas. Determina la viabilidad del proyecto desde el punto de vista empresarial.
Fase de elaboración	Desarrollo de prototipos arquitectónicos	Completa el análisis y el diseño para todos los requisitos de alto riesgo. Desarrolla el prototipo arquitectónico.	Fase de elaboración	Se aclararon las cuestiones arquitectónicas. Riesgos técnicos mitigados. Prototipo inicial para revisión del usuario.
Fase de construcción	Desarrollo	Implemente y pruebe los requisitos clave de 1 para proporcionar la versión 1. Evalúe si la versión está lista para su	Capacidad operativa inicial (código 1 completo)	Todas las características clave desde una perspectiva arquitectónica y de usuario implementada

Fase	Iteración	Descripción	Hitos asociados	Riesgos abordados
		prueba-.		
Fase de transición	Desarrollo / implementación de la versión 1	Implemente la versión 1. Corrija los defectos 1 e incorpore comentarios 1 Implementar y probar los requisitos restantes de 1 . Empaquete, distribuya e instale la versión 1. Los casos de uso de 2 de bajo riesgo restantes están completamente detallados.	Prueba 1 completada Código 1 completo Lanzamiento del producto 1	Comentarios de los usuarios antes del lanzamiento de 1. La calidad del producto debe ser alta. Defectos minimizados. Reducción del coste de la calidad. La liberación en dos etapas minimiza los defectos. La versión en dos etapas proporciona una transición más fácil para los usuarios. Revisado completamente por la comunidad de usuarios
	Desarrollar 2 interno 1	Diseñar, implementar y probar los requisitos de 2 Internal 1. Incorpore mejoras y defectos de 1. Implemente 2 Internal 1.	2 prueba interna 1 completa	Si es necesario, 2 Internal 1 podría liberarse para abordar los defectos de 1, para ayudar a abordar la satisfacción del cliente.
	Desarrollo R2 interno 2	Diseñar, implementar y probar los requisitos de 2 Internal 2 Incorpore mejoras y defectos de 2 Internal 2. Implemente 2	2 Prueba interna 2 completa	2 Internal 1 revisado informalmente por la comunidad de usuarios. Si es necesario, 2 Internal 1 podría liberarse para abordar los defectos de 1,

Fase	Iteración	Descripción	Hitos asociados	Riesgos abordados
		Internal 2.		para ayudar a abordar la satisfacción del cliente.
	Desarrollo / implementació n de la versión 2	Empaquete, distribuya e instale 2 Release.	Código 2 completo Lanzamiento del producto 2	2 Internal 2 revisado informalmente por la comunidad de usuarios. La versión en dos etapas proporciona una transición más fácil para los usuarios.

Table 6.3 Iteraciones junto con los hitos asociados

Además, habrá lanzamientos internos, para mantener un latido regular para ayudar a mantener el proyecto en marcha y permitir la posibilidad de lanzamientos adicionales después del lanzamiento inicial, si es necesario. Los comunicadosinternos pueden ser revisados.

5.2 Calidad de código

No sobrecargar los métodos
Comprobar que los métodos hacen lo que tienen que hacer y no desarrollan una excesiva funcionalidad que provoca que su mantenimiento sea complejo. No es recomendable crear métodos con más de 100 líneas de código

No sobrecargar las clases
No es recomendable crear clases con excesiva funcionalidad. Si una clase está compuesta de excesivos métodos es más difícil realizar trazas sobre los errores. No es recomendable asociar más de 10 métodos a una clase o realizar clases con másde mil líneas de código

Controlar el número de parámetros en una llamada
Un número excesivo de parámetros indica que pueden envolverse mediante el uso de objetos. No es recomendable realizar llamadas que involucren a más de diez parámetros

No utilizar la función eval()
La función eval( ) se utiliza para evaluar string en PHP . Por ejemplo:

<?php

```
$name = 'Chris';
$string = 'echo "Hello, $name";';
eval($string);
?>
```

Este código ejecuta $string como si fuera PHP , lo que es equivalente al código siguiente:

```
<?php
$name = 'Chris'; echo
"Hello, $name";
?>
```

Aunque es útil, eval( ) es muy peligroso si se utiliza con datos manipulados. Por ejemplo si usa una variable $name que haya sido vulnerada, un atacante puede ejecutar código arbitrario de PHP:

```
<?php
$name = $_GET['name'];
eval($name);
?>
```

Es recomendable prohibir el uso de la función eval() cuando sea posible y cuando no lo sea, asegurar que los datos usados en la construcción no han sido manipulados.

Desactivar la función phpinfo

La función phpinfo () produce una página de información sobre la versión PHP está funcionando, cómo está configurado, y así sucesivamente. Debido a que la salida de phpinfo () produce tanta información, es recomendable restringir el acceso a cualquier recurso que utiliza esta función.

Hay que asegurar que nunca se muestra la salida de phpinfo () para el público, ya que expone el contenido de la matriz $_SERVER , lo que facilitaría información para proyectar ataques sobre nuestra aplicación

PHP dispone de una directiva de configuración, que podemos activar a través del php.ini para deshabilitar aquellas funciones del lenguaje que puedan poner en riesgo la seguridad del sistema. La variable de configuración que estamos hablando es disable_functions, a la que asignamos separadas por comas las funciones que puedan ser peligrosas y que, si estamos seguros de no utilizar, es mejor que esténdesactivadas.

Simplemente tenemos que localizar el php.ini de nuestro sistema, que se puede vera través de la función phpinfo(), y editar la variable disable_functions, que estará probablemente sin ninguna función asignada.

disable_functions = phpinfo
En el momento que estén desactivadas, si cualquier persona las utiliza recibirá un mensaje como este:

Warning: phpinfo() has been disabled for security reasons in C:wampwwwinfophp.php on line 4

No usar die() para el manejo de errores
En algunas ocasiones utilizamos la función die() en condicionales para realizar un manejo sobre errores. Esto puede suponer graves problemas para la aplicación, enel caso de que se produzca el error no se reportara información del mismo provocando una sensación poco agradable en el usuario. Es más recomendable utilizar funciones como trigger_error() y en conjunto con set_error_handler() para manejar los errores de tu aplicación.

*No usar funciones dentro de bucles de conteo, precalcular el tamaño de los mismos*Este código obvio que funciona, pero el problema es que la función count() que es ejecutada dentro de la iteración del bucle torna penaliza severamente la velocidad del proceso el proceso.

```
<?php
for ($i = 0; $i < count($miArray); $i++)
{
   // Código
}
 ?>
```

Si calculamos la función fuera del bucle, mejoramos la velocidad en torno a un 600%con respecto a la función inicial.

```
<?php $total = count($miArray);
for ($i = 0; $i < $total; $i++)
{
   // Código
}
 ?>
```

Declarar los métodos estáticos cuando sea posible
Si es posible, hay que declarar los métodos como estáticos si van a ser tratados de esta manera. Está probado que se reduce su tiempo de ejecución hasta cuatro veces con respecto a los métodos que no están declarados como estáticos.

Utilice la variable $_SERVER ['REQUEST_TIME'] sobre la función time()

Cuando necesite calcular el tiempo actual dentro de la ejecución del algún script dentro de una aplicación es mucho más eficiente que utilizar $_SERVER ['REQUEST_TIME'] que la función time().

Utilice foreach() en los bucles de colecciones y arrays

Cuando tenemos una estructura con un bucle destinado a la lectura de un array o una colección está demostrado que es mucho más rápido utilizar foreach() ,preferentemente a estructuras del tipo while o for.

```
// Un ejemplo basado en array Hash con 100 elementos, 24 byte para claves y 10kdatos por cada entrada
foreach($aHash as $val) // Tiempo de ejecución + 100 %
while(list(,$val) = each($aHash) // Tiempo de ejecución + 579 %
```

Utilice strpos() para las búsquedas de subcadenas

Para realizar la búsqueda de subcadenas dentro de cadenas de texto, la mejor manera de realizarlo es utilizar la función strpos(), preferentemente sobre preg_match() o ereg().

```
<?php

if (strpos($authors, 'Chris') !== FALSE) {
 echo 'Chris is an author.';
} else {
echo 'Chris is not an author.';
}

?>
```

Realice preincremento de las variables cuando sea posible

Realizar un preincremento como ++$i es más rápido como el postincremento $ i++en las variables, así que use el preincremento cada vez que es posible. El preincremento es de casi 10% más rápido, lo que significa que usted debe cambiar de las instrucciones para que hagan el pre-incremento cuando se tiene la oportunidad, sobre todo en los bucles críticos.

Realizar depuraciones de código

Un problema con PHP es que, por defecto, si encuentra mensajes de error no fatalessólo se les de salida junto con el resto de su producción, lo que significa que muy amenudo no se dan cuenta de los errores.

En el mundo de la programación, una regla es bastante constante: el código se ejecutará con rapidez hasta que se tiene que se tienen que manejar los errores. Es decir, errores en el código pueden consumir los recursos dedicados

Usted debe revisar cuidadosamente la salida que sus páginas producen con el fin de asegurarse de que PHP no está emitiendo los errores a sus espaldas. Como alternativa, puede asegurarse de que el error de registro está activado en su archivo php.ini, y comprobarlo con regularidad.

Estas funciones aumentan la capacidad de depuración, permitiendo comprobar los scripts sin necesidad de mostrar datos en HTML. Algunas de las más conocidas son APD,Xdebug y ZendStudio.

Usar conexiones persistentes

Si conecta con la base de datos, es interesante considerar el uso de conexiones persistentes más que el uso de conexiones normales. Para usuarios de MySQL, es la diferencia entre usar mysql_pconnect() mejor que mysql_connect().

Las conexiones persistentes permanecen conectadas, aunque el script haya terminado, lo que significa que el siguiente script pregunte por una conexión, usa la única que está disponible. Esto ahorra mucho tiempo negociando las contraseñas y ahorra la ejecución de una parte importante de código.

Para convertir una conexión en persistente no necesita más cambios que añadir una "p" en el nombre de la función, manteniendo los mismos parámetros.

6. Conclusiones

- La definición de un proceso en un proyecto de software es un elemento fundamental para obtener un producto de software de calidad adecuada.
- El plan de calidad es fundamental durante todo el ciclo de vida de desarrollo de un software.
- Una vez que se ha documentado el plan de la calidad para un proyecto es importante tener en cuenta que el mismo deberá ser mantenido conforme este avanza, adelantando siempre su elaboración a la realización de las actividades y adecuando la forma de trabajo para reflejar el aprendizaje del equipo, buscando evitar la ocurrencia de problemas.

Si este documento se encuentra impreso no se garantiza su vigencia, por lo tanto es copia No Controlada. La versión vigente reposará en la intranet institucional. Verificar su vigencia en el listado maestro de documentos.