*Facultad 4*

# *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas*

***Título:*** *Perfil de instalación para el sistema de gestión multieventos*

***Autores:*** *Yaritza Contrera Contino*

*Adrian Quesada Gómez*

***Tutores:*** *MSc. Licet Gutiérrez Mompié*

*Ing. Diego Miguel Torres Salazar*

*Cuidad de La Habana, Cuba*

*"Año 58 de la Revolución"*

Declaramos que somos los únicos autores de este trabajo y autorizamos al Centro FORTES de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Autor(es):

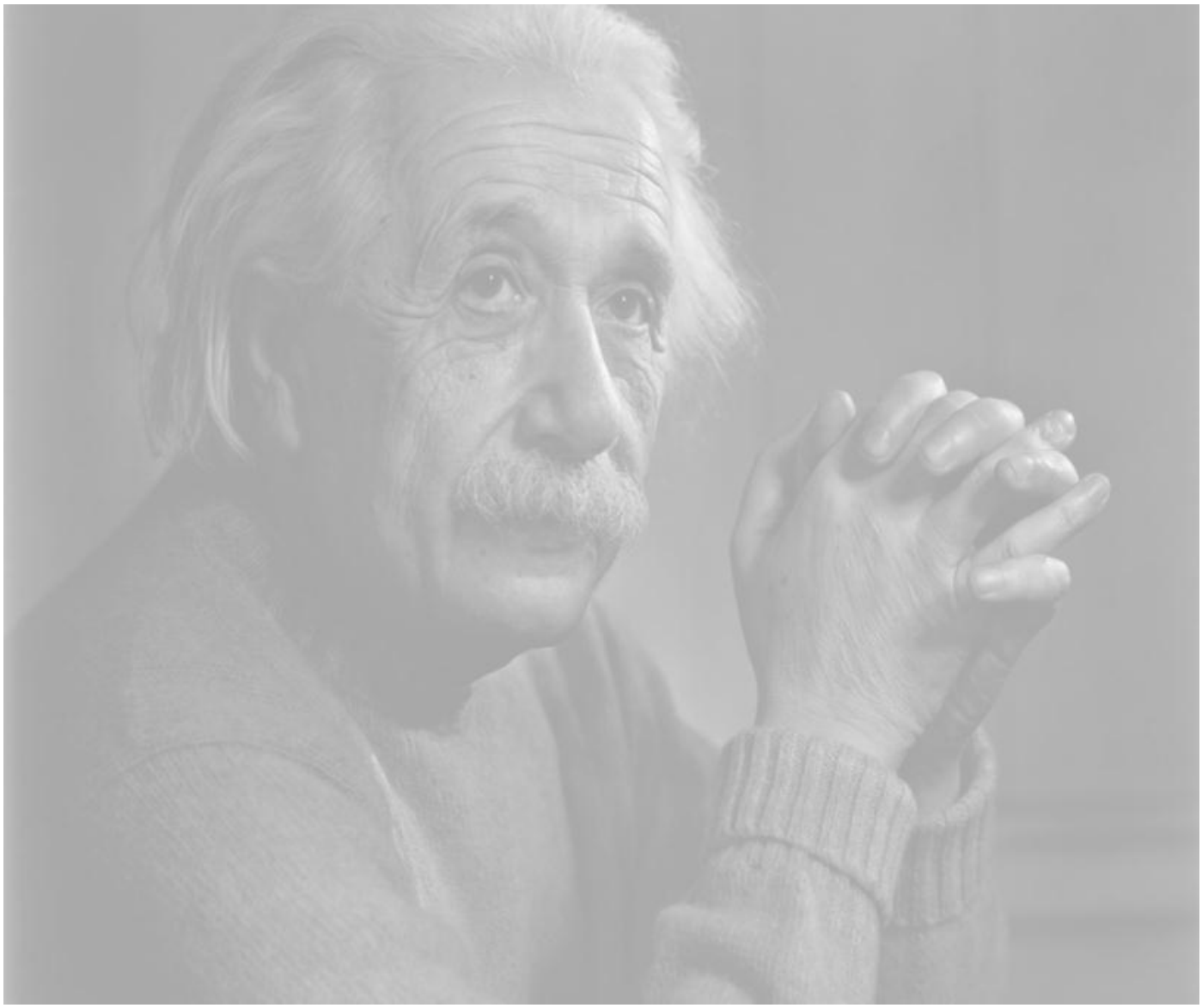
_____ _____

Yaritza Contrera Contino Adrian Quesada Gómez

Tutor(es):

_____ _____

MSc. Licet Gutiérrez Mompié Ing. Diego Miguel Torres Salazar



*“Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad”.*

*Albert Einstein*

*Dedico este trabajo a mi familia por ser mi faro de inspiración, el motor que me impulsa a seguir día tras día. A mis amigos, a todos aquellos que en un momento u otro estuvieron a mi lado y compartimos momentos inolvidables. A mis profesores por haber compartido sus conocimientos y valores para convertirme en mejor persona. A todos ellos gracias.*

*Agradezco a mis padres María del Carmen Contino Blanco y Felix Contrera Yong por apoyarme siempre en todas mis decisiones, por su infinito amor y por siempre ser mis ejemplos a seguir. Gracias por luchar junto a mí para alcanzar mis metas y lograr mis sueños, gracias una y mil veces por su entrega incondicional y por siempre estar pendiente de mí.*

*Agradezco a mi padrastro Andrés Rojas Duarte por ser un segundo padre para mí, por velar siempre por mi bienestar y mi felicidad.*

*A mis hermanos Anadelis y Yasiel para los cuales siempre he intentado ser una fuente de confianza y de inspiración. Gracias por estar siempre conmigo.*

*A mis abuelos por confiar siempre en mí y por darme todo su amor, por estar siempre presente en cada decisión de mi vida. Gracias por siempre estar pendiente de mi felicidad y mi salud.*

*A mis tías y primas por quererme tano, por guiarme siempre por el buen camino y aconsejarme en todo momento.*

*Agradezco a mis tutores por apoyarme, por siempre confiar en mí y guiarme por el camino correcto para lograr el fruto de este trabajo.*

*A todos los profesores que durante estos cinco años me han ayudado a formarme como profesional y como una mejor persona. A todas aquellas personas que a lo largo de mi carrera me apoyaron en todo momento.*

*A todos mis compañeros de clase, en especial a los que han estado pendientes de la evolución y éxito de este trabajo.*

*A mis compañeras de apartamento por ser más que mis amigas, mi segunda familia dentro y fuera de la escuela. Gracias por brindarme siempre su apoyo incondicional en los momentos buenos y malos a lo largo de estos maravillosos años juntas.*

*A mis amigas de la infancia y adolescencia por estar siempre conmigo a pesar de la distancia.*

*A todas aquellas personas que han formado parte de mi vida, gracias por ser en todo momento mi refugio, mi sostén y mi reserva de energías para poder seguir adelante y superar cualquier obstáculo. Gracias a*

*todos ustedes por tener un espacio en su corazón para mí, por cada día tratar de sacarme una sonrisa y andar siempre de la mano de cada instante de mi vida. Los quiero muchísimo.*

*Yaritza Contrera Contino.*

*A mis padres Ivis Gómez Vera y Rodobaldo Quesada Pozo, por el apoyo, el ejemplo, por ser los precursores de toda mi vida, sin ellos nada de esto hubiese sido posible. También a mi madrastra Elisa, que para mí ha sido como una segunda madre y de corazón le agradezco todo lo que ha hecho por mí.*

*A mis abuelos Lusia, Rodobaldo y Magalis por haberme criado y estar siempre ahí para mí, ya que son mi luz en la oscuridad, para ellos va este triunfo. A mi bisabuela Nine porque a pesar de su avanzada edad siempre hacia todo lo posible por mantenerse al tanto de lo que acontecía con mi carrera.*

*A mi hermano Andy, al cual le deseo lo mejor y espero que la vida le brinde la posibilidad de realizarse como hombre de bien.*

*A mi tío Gerardo, mi tía Clara, mis primas Beatriz y Melisa y al resto de la familia por brindarme su apoyo incondicional y estar presentes en los momentos más importantes de mi vida.*

*A mi novia y amiga Liliana por su apoyo, afecto y comprensión durante todo el desarrollo de la tesis. También a su familia, en especial su abuela Ofelia, su mamá Niurys y su hermano Ramón, todas excelentes personas que siempre estuvieron al tanto de mí durante el desarrollo de este trabajo.*

*A mis tutores por compartir su tiempo y conocimiento en aras de lograr este resultado, realmente valoro su disposición ya que han sido una guía profesional invaluable. También a la compañera Liliana que a pesar de no ser tutora oficial siempre aportaba su conocimiento para contribuir con el desarrollo de este trabajo. No puede faltar mi compañera de tesis Yaritza por soportarme durante todo este tiempo.*

*A mis amigos de toda la carrera que siempre estuvieron presentes en las buenas y en las malas, Henry, Rainer, Aynel, Arletty y todos los demás que nos los menciono porque son muchos, pero siempre los tengo presentes.*

*A mis compañeros de aula que durante estos 5 años hemos compartido nuestras vidas pasando por momentos inolvidables debatidos en largas noches de estudio, discusiones, risas, dificultades y alegrías. Daryl, Yusniel, César, Liuba, Ángel, Alexander, Danger, Reinier, y un agradecimiento especial al amigo Frank que a pesar de estar ocupado con su tesis no escatimó en sacrificar parte de su tiempo para brindarnos su ayuda.*

*Por último y no menos importante, agradezco a todos los profesores que contribuyeron con mi formación profesional, en especial a los profesores Marcel, Feliberto, Lara, Pompi, Mailín, Karenia, Yasiris, Yorgelis y Arcel cuya ayuda fue de vital importancia para la investigación.*

*Adrian Quesada Gómez.*

## Resumen

El Centro de Desarrollo de Tecnologías para la Formación (FORTES) de la Universidad de las Ciencias Informáticas desarrolla actualmente un sistema para la gestión de múltiples eventos científicos. La vía mediante la cual se instala actualmente dicho producto requiere mucho esfuerzo en cuanto a la realización de las tareas de configuración, por lo que surge la necesidad de desarrollar una aplicación que realice estas acciones en los entornos de despliegue. El presente trabajo de diploma presenta un estudio de las características de diseño y desarrollo de diferentes perfiles de instalación, así como las tecnologías usadas, las herramientas y principales conceptos para la creación de un perfil de instalación. Igualmente aborda sobre el proceso de desarrollo del producto implementado y los aspectos técnicos del mismo, así como la validación de la solución propuesta.

Se describe generalmente el desarrollo de un perfil de instalación para el Sistema de Gestión Multieventos (SGM), con el uso del Sistema de Gestión de Contenidos (CMS) Drupal en su versión 7, incluyendo características para facilitar los procesos de instalación y configuración de dicho sistema y minimizar el esfuerzo empleado en el despliegue del mismo.

**Palabras clave:** drupal, instalación, perfil, sistema de gestión de contenidos, sistema de gestión multieventos.

**Índice**

## Introducción

Actualmente las universidades cubanas centran sus estrategias en potenciar cada vez más la línea de producción científica para la formación integral de estudiantes y profesores manteniendo como objetivo principal el desarrollo de aplicaciones que posibiliten la libre publicación de los trabajos presentados en eventos científicos. Siguiendo esas líneas para la formación integral se encuentra inmersa la Universidad de las Ciencias Informáticas (UCI) como centro de enseñanza superior; la cual aspira a formar profesionales de la informática con un alto nivel científico. Para ello se han creado múltiples centros productivos; entre los que se encuentra el Centro de Tecnologías para la Formación de la Facultad 4, este contempla entre sus proyecciones el desarrollo de una aplicación para la gestión de eventos científicos.

El mismo permite a partir de un único sistema el manejo de varios eventos científicos. Guarda las evidencias pertenecientes a la participación en eventos de este tipo, para la promoción y difusión de los resultados alcanzados en cada uno de ellos. En cuanto a los procesos para su instalación, configuración y despliegue presenta algunos problemas los cuales imposibilitan la personalización del producto. Las limitantes se enumeran a continuación:

- Las tareas de instalación y configuración se realizan manualmente debido a que es necesario:

✓ Copiar manualmente hacia el servidor web en cuestión la carpeta con el código fuente del núcleo de Drupal en su versión 7 y los archivos pertenecientes al proyecto (cachés, módulos personalizados y librerías).

✓ Importar el SQL de la base de datos manualmente a través del gestor de base de datos PostgreSQL.

✓ Realizar las configuraciones relacionadas con (Puerto, Nombre de la base de datos(BD), Contraseña de la BD, Usuario de la BD) en el fichero de configuración settings.php, haciendo de este un proceso engorroso y consumiendo mucho tiempo.

- El administrador del sistema debe tener un amplio conocimiento del funcionamiento del Sistema de Gestión Multieventos(SGM) para poder instalarlo.
- La vía de instalación actual dificulta el proceso de despliegue del producto, perjudicando así la posible comercialización y mantenimiento del mismo.

Teniendo en cuenta lo antes descrito, se formula el siguiente problema de investigación: ¿Cómo contribuir al proceso de instalación y configuración del SGM, para minimizar el esfuerzo en cuanto al despliegue?

Se tiene como objeto de estudio: perfiles de instalación para el CMS Drupal en su versión 7. Enmarcado en el campo de acción: perfiles de instalación para el SGM con el uso del CMS Drupal en su versión 7, se establece como objetivo general: implementar los procesos de instalación y configuración del SGM; para minimizar el esfuerzo en cuanto al despliegue.

Para la realización de dicho perfil de instalación se plantean las siguientes preguntas científicas:

- ¿Qué características comunes poseen las soluciones homólogas existentes?
- ¿Cómo se realiza actualmente el proceso de instalación del SGM?
- ¿Cuáles son las herramientas y tecnologías necesarias para el desarrollo de perfiles de instalación?
- ¿Cuánto esfuerzo es necesario para la configuración, instalación y despliegue del SGM actualmente?
- ¿Qué tipo de pruebas de software se aplican para validar la calidad de un instalador?

Para dar cumplimiento a estas preguntas científicas se establecen los siguientes objetivos específicos:

- Construir el marco teórico referencial mediante la consulta, extracción y recopilación de la información relevante sobre el tema de investigación.
- Analizar y diseñar un perfil de instalación para el SGM.
- Desarrollar la solución propuesta.
- Implementar el perfil de instalación.
- Realizar mediciones en cuanto al esfuerzo empleado durante la instalación del SGM.
- Validar el producto desarrollado.

Para el cumplimiento de dichos objetivos se desarrollarán las siguientes tareas de investigación:

- Estudio de las soluciones similares al problema a resolver.
- Análisis y diseño para el posterior desarrollo del perfil de instalación.
- Definición de los requisitos funcionales y no funcionales de la propuesta de solución.

- Desarrollo de la propuesta de solución.
- Aplicación de pruebas al producto desarrollado.

Para el cumplimiento de las tareas de la investigación, se emplean los siguientes métodos investigativos:

Métodos teóricos:

Inductivo-Deductivo: se empleó para el planteamiento del objetivo general y la definición de algunos requisitos que debía tener el perfil de instalación, permitiendo así llegar a conclusiones lógicas a partir de los conocimientos adquiridos.

Análisis Histórico-Lógico: se utilizó para el análisis de la evolución de las herramientas para la creación de los perfiles de instalación, además del estudio de las novedades que ha tenido el proceso de desarrollo de perfiles a través de las diferentes versiones del CMS Drupal.

Analítico-Sintético: permitió seleccionar los elementos más importantes relacionados con los perfiles de instalación.

Métodos Empíricos:

Entrevista: utilizada para definir características, cualidades y gran parte de los requisitos funcionales que debía tener el perfil de instalación.

El presente documento está estructurado por los siguientes capítulos:

**Capítulo 1: Marco teórico-conceptual para el desarrollo del perfil de instalación**

En este capítulo se abordarán elementos teóricos que apoyan la investigación como, por ejemplo: conceptos fundamentales, herramientas, tecnologías, actual proceso de instalación y configuración del SGM, así como un estudio a varios perfiles de instalación y sus conceptos asociados. Se reflejan además elementos del CMS Drupal y su integración con perfiles de instalación.

**Capítulo 2: Propuesta de solución y características del perfil de instalación**

En este capítulo se exponen los detalles correspondientes a la propuesta de solución, se describe el flujo de los procesos involucrados y se detallan las características propias del perfil de instalación desarrollado. Además, se explican las actividades de ingeniería y gestión de software llevadas a cabo durante el desarrollo del mismo.

**Capítulo 3: Implementación y prueba del perfil de instalación**

En este capítulo se exponen los estándares de codificación aplicados al código del perfil de instalación, además de explicar el tipo de prueba aplicada a la solución planteada y realizar un análisis de los resultados obtenidos para comprobar su correcto funcionamiento.

# *Capítulo 1*

## *Marco teórico-conceptual para el desarrollo del perfil de instalación*

## Introducción

Para llegar a comprender el estudio realizado en una investigación es necesario conocer los conceptos fundamentales que la sustentan, conformando así el marco teórico conceptual o fundamentación teórica del mismo.

## 1.1 Conceptos asociados a la investigación

En las últimas décadas la industria de la informática y las comunicaciones ha escalado peldaños en la sociedad, adentrándose en los más variados escenarios como consecuencia de las crecientes necesidades de las empresas y organizaciones. Los sistemas de gestión han contribuido en gran parte a ese avance, agilizando el trabajo y permitiendo obtener mayores resultados en las actividades productivas.

**Sistema de Gestión Multieventos(SGM):** Sistema desarrollado en el centro FORTES de la facultad 4 para la gestión de múltiples eventos científicos. Tiene como objetivos principales:

- Controlar y organizar de forma digital los eventos.
- Revisión, evaluación, localización, reutilización y recuperación de los trabajos presentados en los eventos.

Fue desarrollado con el uso del CMS Drupal en su versión 7. A continuación, se muestra la página principal del SGM, disponible en: forum.eventos.mes.gob.cu


Figura 1. Página de inicio del SGM.

Los avances tecnológicos en internet han revolucionado la forma de desarrollo y administración de sitios web, los métodos ágiles para el desarrollo web surgen precisamente como reacción ante los métodos tradicionales, inicialmente las aplicaciones o sistemas web eran realizados completamente con programas como Macromedia Dreamweaver, estas herramientas estaban enfocadas más a la creación que al mantenimiento. Posteriormente a estos programas, han aparecido los Sistemas de Gestión de Contenidos (CMS) para agilizar el desarrollo de portales o sitios web, los que han adquirido mucha aceptación.

*"**CMS**: Sistema de gestión de contenidos o CMS de las siglas del idioma inglés que significan: Content Management System."A principios de los años noventa, el concepto de sistemas de gestión de contenidos era desconocido. La empresa enfocada a la Web fue una de las empresas pioneras que empezó el desarrollo de un gestor de contenidos en el año 1994. No fue hasta a finales del año siguiente que presentaron su CMS basado en una base de datos"*. (Sirilo Juno 2013; Raúl 2013)

Los CMS de código abierto empezaron su desarrollo el año 1997, en palabras de su autor, Kasper Skårhøj, la herramienta que popularizó el uso de estos sistemas para las comunidades de usuarios en Internet, se empezó a desarrollar en el año 2000.

Según Sirilo Juno *,"un CMS es un programa que nos sirve para crear, editar, gestionar, publicar y administrar contenidos".(Sirilo Juno 2013)* (Dries Buytaert 2015)

1. Brinda una interfaz simple y fácil de usar.
2. Permite mayor dinamismo y colaboración entre los participantes del proyecto.
3. Ahorro de costos y tiempos en el desarrollo del proyecto.
4. Colaboración de distintas comunidades afines al Sistema de Gestión de Contenido, lo que facilita el adherir plugins y complementos.
5. Permite cambiar de interfaz con la simple actualización de un tema, este le dará la apariencia al proyecto.

Entre los CMS de código libre se destacan, Drupal, Joomla y WordPress.

*"**Drupal** está escrito en lenguaje PHP y se basa en una implementación base, más conocida como "Drupal Core", que contiene los elementos básicos para la creación de un sitio web bajo este sistema."* (José A. Senso, 2010) (Fran Gil Rodríguez 2011b, 2011a)

Según la página oficial de Drupal disponible en (https://www.drupal.org/)**,** este CMS se utiliza para crear sitios web dinámicos y con gran variedad de funcionalidades. Es un software libre, escrito en PHP, surge en enero de 2001, su creador fue Dries Buytaert el cual decidió liberar el software tras drop.org como Drupal."(Dries Buytaert 2015; Fran Gil Rodríguez 2011b, 2011a)

Con el uso de este CMS es posible implementar una gran variedad de sitios web: un blog personal o profesional, un portal corporativo, una tienda virtual, una red social o comunidad virtual. (Rochen 2015; Raúl 2013; Dries Buytaert 2013, 2009). Dicho "Core" cuenta con componentes primarios de Drupal, que incluyen capacidades para crear y manejar:

- Contenido
- Archivos

- Menús
- Cuentas de usuarios
- Permisos y roles de usuarios
- Categorización (taxonomías)
- Apariencia
- Módulos

Los módulos obligatorios que forman parte del núcleo de Drupal en su versión 7 son:

- Field y Field SQL storage: permiten conjuntamente añadir campos a entidades, tales como nodos y usuarios.
- Filter: realiza acciones de filtrado sobre los contenidos a mostrar.
- Image: permite al sistema manejar imágenes.
- Node: realiza las acciones necesarias para la publicación y gestión.
- System: se encarga de la administración general del sitio.
- Taxonomy: permite la categorización de contenido.
- Text: define tipos de campo de texto simple.
- User: es necesario para el registro, acceso y gestión de usuarios.

Estos módulos son obligatorios y dado que Drupal no podría funcionar sin ellos, no es posible prescindir de los mismos. (Fran Gil Rodríguez 2011a, 2011a)

Drupal durante su proceso de instalación ejecuta un conjunto de tareas o instrucciones las cuales permiten la activación de los módulos, temas, opciones y configuraciones requeridas por el CMS para su funcionamiento. Estas tareas o instrucciones son establecidas o definidas en un perfil de instalación

Después haber sido consultadas varias bibliografías((Hong Pong Japerry 2015), (Fran Gil Rodríguez 2011b, 2011a)) se define el concepto de perfil de instalación según la página oficial de Drupal disponible en (https://www.drupal.org/)**:** *"Un **perfil de instalación** es un conjunto de módulos y/o temas, que se*

*instalan y se configuran de una forma determinada durante el proceso de instalación.* (Dries Buytaert 2015)

Los perfiles de instalación son una manera de personalizar el proceso de instalación de Drupal, se puede agregar a la instalación muchas opciones como definir formularios, procesos o pasos adicionales, habilitar nuevos módulos y crear bloques. Un perfil de instalación es útil usarlo cuando se repiten las mismas tareas para el desarrollo de sitios con regularidad. Estas tareas pueden ser automatizadas sin necesidad de empezar de cero el desarrollo, ganando de esta forma en tiempo y productividad por parte de los desarrolladores.

Basado en todo lo antes expuesto sobre los perfiles de instalación se arriba a la siguiente conclusión:

Con un perfil de instalación se puede configurar un sistema completo con las menores configuraciones posibles realizadas por el usuario, permitiéndole con pocos datos introducidos los mejores servicios para el uso del sistema y en todo momento hacerle saber el proceso que se está efectuando por pasos y barras de progreso, ganando así en tiempo y esfuerzo de desarrollo. Un perfil de instalación es una buena opción para la creación de una distribución de Drupal o para situaciones en las que se construye un mismo sitio web repetidamente.

## 1.2 Tipos de perfiles de instalación

El CMS Drupal provee 3 perfiles de instalación, por defecto ofrece el Minimal (Mínimo) y el Standard (Estándar). La versión Standard instalará más módulos que la versión Minimal. El tercer perfil, llamado Testing, está definido como oculto y no se muestra como seleccionable durante el proceso de instalación de Drupal.

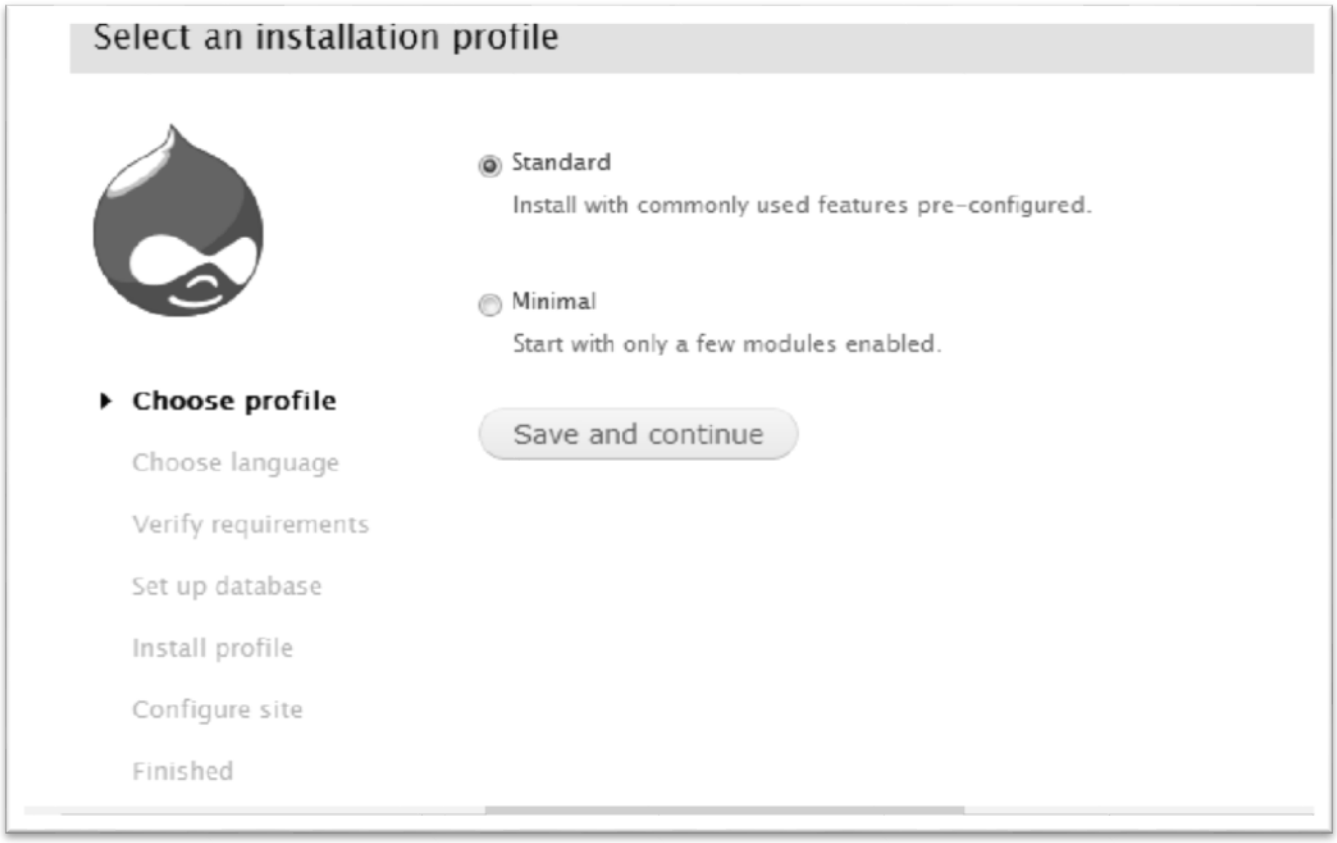



Figura 2. Selección de perfiles en Drupal.(Fran Gil Rodríguez 2011b)

El perfil Standard instala y configura algunas de las funcionalidades básicas generalmente utilizadas para el desarrollo de cualquier sitio web. Permite habilitar varios módulos básicos, herramientas de administración por defecto, así como establecer una configuración inicial necesaria. Este perfil de instalación es el recomendado para instalar, ya que muestra lo básico que se puede hacer en cualquier sitio web, pudiéndose ahorrar tiempo de creación por tener configuraciones por defecto para los casos de desarrollo más comunes.

El perfil de instalación Minimal (Mínimo) instala solo unos módulos muy básicos. Este perfil es útil si sólo se desea tener características muy específicas, o si se necesita más trabajo personalizado como para deshacer las configuraciones predeterminadas proporcionadas por el perfil Standard(Dries Buytaert 2015). Todos estos perfiles de instalación mantienen una misma estructura interna.

## 1.3 Estructura de archivos del perfil de instalación

Un perfil de instalación se compone típicamente, por los siguientes archivos y carpetas:

- Archivo .info: archivo de definición del perfil de instalación. Su contenido es similar al archivo .info de módulos y temas. Ejemplo del contenido de este archivo:

```
name = Mi distribución
description = Es una distribución de ejemplo
core = 7.x
dependencies[] = block
dependencies[] = color
dependencies[] = comment
dependencies[] = contextual
dependencies[] = dashboard
dependencies[] =...
files [] = mi_distribución.profile
```

- Archivo .profile: Es el archivo donde se implementan las tareas que se llevarán a cabo durante la instalación. Es equivalente al archivo .module de los módulos y en él se podrán usar la mayoría de las funciones disponibles de la API de Drupal. Ejemplo del contenido de este archivo:

```
**
* Implements hook_install_tasks().
*/
function mi_distribucion_install_tasks() {
 $areas = array();
 $tareas['mi_distribucion_settings_form'] = array(
```

```
 'display_name' => st('Additional options'),
 'type' => 'form',
 );
 return $tareas;
}
```

Para modificar la tareas de instalación de Drupal, se utiliza el **hook_install_tasks_alter().**

- Archivo .install: Permite implementar hook_install () para insertar contenido en la base de datos durante la instalación. Ejemplo del contenido de este archivo:

```
/**
* Implements hook_install().
*/
function mi_distribucion_install() {
 include_once DRUPAL_ROOT. '/profiles/standard/standard.install';
 standard_install ();
 //nuestro código...
}
```

- Carpeta /modules: en esta carpeta se pueden añadir módulos que estarán disponibles durante la instalación. Generalmente sólo se usará esta carpeta para añadir módulos creados específicamente para la distribución. El resto de módulos requeridos serán registrados como dependencias, de forma que serán solicitados durante la instalación.
- Carpeta /themes: en esta carpeta se pueden añadir temas que estarán disponibles durante la instalación. Generalmente sólo se usará esta carpeta para añadir temas creados específicamente para la distribución.
- Carpeta /translations: en esta carpeta se añaden las traducciones del núcleo, necesarias para que el proceso de instalación se realice en otro idioma (por ejemplo, en español). (Dries Buytaert 2013)

## 1.4 Estudio de homólogos

Se pueden crear además de los 3 perfiles que existen por defecto en el CMS Drupal otros perfiles de instalación, los cuales sean capaces de ajustarse a la temática de desarrollo en cuestión y a las necesidades que existan. Algunos ejemplos de ellos son evidenciados en las distribuciones que se muestran a continuación:

- **Drupal commons:** permite crear comunidades de usuarios donde es posible, organizar blogs, wikis, crear interacción entre miembros de una comunidad, instala tipos de contenidos, define sistema de taxonomía y posee temas adaptados.
- **OpenPublish:** orientado a la creación de un periódico digital, instala alrededor de 50 módulos contribuidos permitiendo agregar nuevas funcionalidades al sitio, instala siete tipos de contenidos y define un sistema de taxonomía, posee temas adaptados con administración de interfaz elegante e íconos informativos.
- **Opigno LMS:** orientado a la creación de Sistemas de Dirección de Aprendizaje completo basado en Drupal. Permite guiar al estudiante en el proceso de entrenamiento organizado en los cursos y lecciones que matricule, facilita las interacciones gracias a las reuniones presenciales, foros y charlas.
- **Open Outreach:** incluye la última versión del centro de Drupal y otros módulos frecuentemente. Pre configura los rasgos usados por las organizaciones como los calendarios de eventos, imágenes y videos, integración de los medios de comunicación social y dirección del contacto.
- **Commerce Kickstart:** orientado al comercio electrónico (tienda virtual), instala alrededor de 60 módulos adicionales, implementa además alrededor de 20 módulos propios creados usando el módulo “features” con características específicas para el perfil, define tareas adicionales durante el proceso de instalación, instala varios tipos de contenidos, instala contenido de ejemplo para visualizar en el sitio y define sistema de taxonomía. Posee temas adaptados con administración de interfaz elegante. (Dries Buytaert 2015; Raúl 2013; Dries Buytaert 2013).

## 1.5 Características similares de los perfiles de instalación

A continuación, se describen las características similares que poseen los perfiles de instalación:

- Contenido flexible: pueden definir campos personalizados que podrán ser utilizados en tipos de contenido, usuarios, comentarios, términos y otras entidades.
- Accesibilidad: las pantallas de administración son mucho más accesibles. Las abundantes mejoras en el interfaz le facilitan la construcción de productos altamente accesibles.
- Imágenes y ficheros: el soporte de imágenes en el contenido está incorporado en el núcleo, generando versiones diferentes, vistas previas y otros estilos de imágenes. Es posible utilizar las gestiones privada y pública de ficheros al mismo tiempo. Ofrecen un mejor diseño de plantillas y un mayor control de lo que se muestra en pantalla en el momento de hacer modificaciones. (Dries Buytaert 2013; Hong Pong Japerry 2015)

## 1.6 Resultados del estudio realizado a los perfiles de instalación

Los perfiles de instalación estudiados están diseñados para el trabajo y la gestión en un área o solución específica, hacen uso de muchas funcionalidades a considerar, tienen características y opciones útiles, como la creación de blogs, calendarios, gestión y visualización de contenidos, que sirven como modelo y guía para el desarrollo del perfil de instalación del SGM. Son sistemas bien definidos y acoplados, lo que los hace complicados de configurar y adaptar a una solución personalizada.

Se tomarán en cuenta las características del **Commerce Kickstart**, debido a que contiene una apariencia especialmente diseñada para facilitar el proceso de instalación y una vista amigable además de una definición de tareas de instalación óptimas para la solución, lo cual ayuda a enriquecer el producto, logrando obtener la calidad requerida para el uso del mismo por el usuario final y que cumplan con las necesidades requeridas para lograr que el proceso de instalación del SGM sea lo más personalizado posible. No se utilizó directamente el **Commerce Kickstart** debido a que su objetivo no es servir como perfil para un sitio de gestión de eventos científicos.

## 1.7 Entorno tecnológico para el desarrollo del perfil de instalación

Un lenguaje de programación web es aquella estructura que con una cierta base sintáctica y semántica, imparte distintas instrucciones a un programa de computadora. Muchos son los lenguajes que existen actualmente en el mercado de la tecnología y la informática. No obstante, entre los más importantes se encuentran: Pascal, Visual Basic, SQL, Delphi, Lingo, Cobol, HTML o Java. A continuación serán explicados los conceptos asociados a los lenguajes usados para la realización del perfil.(Grupo de desarrollo 2015)

**PHP:** lenguaje de programación, interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+. (PHP Group 2001). Se utilizará el lenguaje PHP en su versión 5.6.17 por ser el lenguaje de desarrollo del CMS.

**CSS3:** hoja de estilo en cascada o CSS (siglas en inglés de *cascading style sheets*) es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El World Wide Web Consortium (W3C) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.(W3C Working Group Note 2015).

El lenguaje CSS sirve para organizar la presentación y aspecto de una página web. Este lenguaje es principalmente utilizado por parte de los navegadores web de internet y por los programadores web informáticos para elegir multitud de opciones de presentación como colores, tipos y tamaños de letra. (Manuel Sierra 2015; Alarcón 2016)

**HTML5:** sigla en inglés de HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros.(W3C Working Group Note 2016)

**JavaScript:** es un lenguaje de scripting multiplataforma y orientado a objetos. Es un lenguaje pequeño y liviano. Dentro de un ambiente de host, puede conectarse a los objetos de su ambiente y proporcionar control programático sobre ellos.(JavaScript community 2015)

## 1.8 Entorno de desarrollo

*"Un entorno de desarrollo integrado(*IDE*), es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI)".(*Bernaldo Flores 2011)

Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, PHP, Python, Java, C#, Delphi, Visual Basic, etc. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto.

**Netbeans:** es un entorno de desarrollo. Herramienta utilizada para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. Se utilizó el mismo en su versión 8.0 como IDE para el desarrollo del perfil de instalación por poseer funcionalidades útiles en el trabajo con el lenguaje de desarrollo PHP.(Oracle Corporation 2016)

## 1.9 Herramientas CASE

*"Se puede definir a las Herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software".*(Evelyn Méndez Alonso 2014)

Algunos ejemplos de herramientas CASE son:

- Visual Paradigm
- Rational Rose
- Erwin

- EasyCASE
- PowerDesigner

**Visual Paradigm:** Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de Software, Analistas de Sistemas y Arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. Se establece como herramienta para el trabajo con la ingeniería en su versión 8.0 ya que se cuenta con una amplia experiencia con el uso del mismo.(Visual Paradigm community 2014)

## 1.10 Proceso de desarrollo de software

*"Una metodología es un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de software detallado y completo. "(Heredia Ruiz et al. 2011; Beastieux Zero 2008)*

Las metodologías se basan en una combinación de los modelos de proceso genéricos (cascada, incremental...). Definen artefactos, roles y actividades, junto con prácticas y técnicas recomendadas. La metodología para el desarrollo de software en un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito. Una metodología para el desarrollo de software comprende los procesos a seguir sistemáticamente para idear, implementar y mantener un producto software desde que surge la necesidad del producto hasta que se cumpla el objetivo por el cual fue creado.

Una definición estándar de metodología puede ser el conjunto de métodos que se utilizan en una determinada actividad con el fin de formalizarla y optimizarla. Determina los pasos a seguir y cómo realizarlos para finalizar una tarea. (Heredia Ruiz et al. 2011)

## 1.11 Metodologías para el desarrollo de software

Desarrollar un buen software depende de un gran número de actividades y etapas, donde el impacto de elegir la metodología para un equipo en un determinado proyecto es trascendental para el éxito del producto.

Según la filosofía de desarrollo se pueden clasificar las metodologías en dos grupos.

Las metodologías tradicionales, que se basan en una fuerte planificación durante todo el desarrollo y las metodologías ágiles, en las que el desarrollo de software es incremental, cooperativo, sencillo y adaptado. (Heredia Ruiz et al. 2011; Tamara Rodríguez Sánchez 2014)

### 1.11.1 Metodologías tradicionales

Las metodologías tradicionales son denominadas, a veces, de forma peyorativa, como metodologías pesadas. Dentro de ellas se encuentran:

- MSF (por sus siglas en inglés Microsoft Solution Framework)
- Métrica 3
- RUP (siglas de Rational Unified Process)

Todas estas centran su atención en llevar una documentación exhaustiva de todo el proyecto y en cumplir con un plan de proyecto, definido todo esto, en la fase inicial del desarrollo del proyecto.

Otra de las características importantes dentro de este enfoque, son los altos costes al implementar un cambio y la falta de flexibilidad en proyectos donde el entorno es volátil. Las metodologías tradicionales (formales) se focalizan en la documentación, planificación y procesos (plantillas, técnicas de administración, revisiones, etc.) (Blum, 2001)(Tamara Rodríguez Sánchez 2014; Beastieux Zero 2008)

### 1.11.2 Metodologías ágiles

Actualmente los métodos ágiles están ganando popularidad en la industria del desarrollo de software debido a que el enfoque ágil posibilita que se hagan ciertos cambios aún en etapas avanzadas del

proyecto, es decir, permite agregar nuevos requerimientos o que estos evolucionen, entregando así el máximo valor posible a la empresa, ya que se reconocen sus necesidades cambiantes.

Este enfoque nace como respuesta a los problemas que puedan ocasionar las metodologías tradicionales y se basa en dos aspectos fundamentales, retrasar las decisiones y la planificación adaptativa. Basan su fundamento en la adaptabilidad de los procesos de desarrollo. Estas metodologías ponen de relevancia que la capacidad de respuesta a un cambio es más importante que el seguimiento estricto de un plan. (Tamara Rodríguez Sánchez 2014; Beastieux Zero 2008)

Dentro de las metodologías ágiles se encuentan:

- Extreme Programming (XP)
- Scrum
- Crystal Methodologies

## 1.12 CMMI (Modelo de Madurez de Capacidad Integrado)

La UCI con el objetivo de mejorar sus procesos y fortalecer su cultura de calidad en el desarrollo de software lleva a cabo la implementación de un proceso de mejora basado en el Modelo de Madurez de Capacidad Integrado (CMMI).

*"CMMI es un modelo de madurez de mejora de los procesos que especifica las mejores prácticas que tratan las actividades de desarrollo de un producto de software. El CMMI le permite a una organización aproximarse a la mejora de procesos y a las evaluaciones usando dos representaciones diferentes, la representación continua y la representación por etapas o escalonada." (Fernández Ladera 2012; Tamara Rodríguez Sánchez 2014)*

Para el desarrollo del perfil de instalación se tiene en cuenta la representación por etapas, al ser este tipo de implementación la definida por la Universidad de las Ciencias Informáticas (UCI).

La representación por etapas ofrece un enfoque sistemático y estructurado para mejorar los procesos paso a paso. Al conseguir cada etapa, se asegura que se ha dado un mejoramiento y que se han establecido las bases necesarias para iniciar la siguiente etapa. Cuenta con 5 niveles de madurez de los procesos, el nivel 2 o gestionado está compuesto a su vez por las siguientes 7 áreas de procesos:

1. Administración de Requisitos.
2. Aseguramiento de la Calidad de Procesos y Productos.
3. Planeación de Proyectos.
4. Monitoreo y Control de Proyectos.
5. Medición y Análisis.
6. Administración de Configuración.
7. Administración de Acuerdos con Proveedores.(Tamara Rodríguez Sánchez 2014)

## 1.13 Fundamentación de la metodología a usar

Se utilizará la metodología AUP-UCI. Su definición se basa en una variación de la metodología "Proceso Unificado Ágil" (AUP por sus siglas en inglés) en unión con el modelo CMMI-DEV v1.3 debido a que con esta variación de la metodología AUP se logra reducir el esfuerzo en tiempo y en personas, converger a los proyectos hacia una sola metodología de desarrollo. Es una metodología para ser adaptada a lo que ya la Universidad ha estado proponiendo como ciclo de vida de los proyectos, sin alejarse de lo que hasta el momento se ha trabajado e introducir la menor cantidad de cambios posibles.(Tamara Rodríguez Sánchez 2014)

Se utilizó el cuarto escenario de la metodología AUP-UCI: Historia de usuario, debido a que en este caso el cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos.

El AUP aplica técnicas ágiles incluyendo:

- Desarrollo Dirigido por Pruebas (test driven development - TDD en inglés)
- Modelado ágil
- Gestión de Cambios ágil
- Refactorización de Base de Datos para mejorar la productividad.

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva.

A continuación, se muestra en una tabla los roles de la variación de AUP UCI con respectos a los roles de AUP.

Tabla 2. Roles definidos en la metodología AUP y AUP-UCI.

| **Roles AUP** | **Roles Variación AUP-UCI** | **Responsabilidades Roles** (Variación AUP-UCI) |
|---|---|---|
| Administrador de proyecto | Jefe de proyecto | Las habilidades y competencias de cada uno de los roles definidos para la Variación de AUP-UCI se pueden consultar en mejoras.prod.uci.cu |
| | Planificador | |
| Ingeniero de procesos | Analista | |
| Modelador ágil | Arquitecto de información. (Opcional) | |
| Desarrollador | Desarrollador | |
| Administrador de la configuración | Administrador de la configuración | |
| Stakeholder | Stakeholder (Cliente/Proveedor de requisitos) | |
| Administrador de pruebas | Administrador de calidad | |
| Probador | Probador | |

**Descripción de las cuatro fases que propone AUP**

De las cuatro fases (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes tres fases de AUP en una sola, definida como Ejecución y se agrega la fase de Cierre**.**(Stevia Rodríguez 2015; Tamara Rodríguez Sánchez 2014)

- **Inicio:** durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
- **Ejecución:** en esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
- **Cierre:** en esta fase se analizan todos los resultados alcanzados.

## 1.14 Conclusiones

El desarrollo del marco teórico de la investigación arrojó resultados importantes para el posterior desarrollo de la solución:

- El análisis de las soluciones similares permitió reconocer la existencia de varios perfiles de instalación. Se realizó un análisis de todas sus características donde se concluye que ningún perfil de los analizados tiene como objetivo principal instalar un sistema de gestión de eventos.
- El estudio de diferentes tecnologías, herramientas y lenguajes de desarrollo posibilitó determinar las que se utilizarían en la elaboración del instalador: AUP-UCI como metodología de desarrollo, UML como lenguaje de modelado, Visual Paradigm en su versión 8.0 como herramienta CASE, PHP en su versión 5.6.17 como lenguaje de desarrollo y NetBeans en su versión 8.0 como IDE de desarrollo.
- De forma general los elementos y características expuestos en el presente capítulo permiten afirmar que su correcto uso posibilitará la construcción de un sistema capaz de resolver la problemática planteada.

# *Capítulo 2*

## *Propuesta de solución y características del perfil de instalación*

## Introducción

Luego de haber definido justificadamente las herramientas y tecnologías utilizadas en el desarrollo del instalador para el SGM se abordarán a continuación los principales rasgos característicos del sistema implementado. También será representada la descripción general de la propuesta del sistema. Además, se relacionarán los requerimientos funcionales y no funcionales de la solución y los artefactos de ingeniería de software generados.

## 2.1 Propuesta de solución

Se propone la implementación de un perfil de instalación para el CMS Drupal en su versión 7 que tenga en cuenta las necesidades del cliente, el objetivo general trazado al inicio de la investigación y los estudios realizados a diferentes perfiles de instalación. Dicha propuesta está basada en facilitar todas las tareas referentes a la instalación y configuración del SGM a través de configuraciones personalizadas.

El perfil de instalación estará desarrollado para ejecutar tareas e instrucciones de configuración que se ejecutarán durante la instalación del SGM. Dichas tareas se mostrarán automáticamente según el orden definido, destacando con color verde las tareas realizadas y con color azul la tarea en cuestión en el momento de la instalación. El proceso de instalación del perfil a desarrollar será guiado por formularios y barras de progreso. A continuación, se presenta la interfaz principal propuesta, la cual muestra una pantalla de bienvenida al usuario.


Figura 3. Interfaz principal del instalador para SGM.

## 2.2 Modelo Conceptual

*"Un modelo conceptual es una representación de conceptos en un dominio del problema. La designación de un modelo conceptual ofrece la ventaja de subrayar fuertemente una concentración en los conceptos del dominio, no en las entidades del software*."(Martin Fowler 2015)

Simplifican lo que se describe de diversas formas, permitiendo a su creador centrarse en los aspectos que considera importantes a modelar. Son instrumentos de reducción de la complejidad dependiendo de la fidelidad con que describa el modelado, se simulará la manipulación de los resultados reales.

A partir de estas características el equipo de desarrollo optó por la realización de un modelo conceptual, con el objetivo de brindar un acercamiento a los principales conceptos del dominio que manejará el instalador, así como una mejor comprensión del actual proceso de instalación.(Martin Fowler 2015; Tamara Rodríguez Sánchez 2014)

De manera general el modelo conceptual es una representación de conceptos del mundo real, no de componentes de software. El objetivo de la creación de este artefacto es aumentar la comprensión del problema y contribuir a esclarecer la terminología o nomenclatura del dominio. Puede verse como un modelo que comunica a los interesados cuáles son los términos importantes y cómo se relacionan entre sí. Se representa mediante un diagrama de clases UML como se puede ver a continuación:

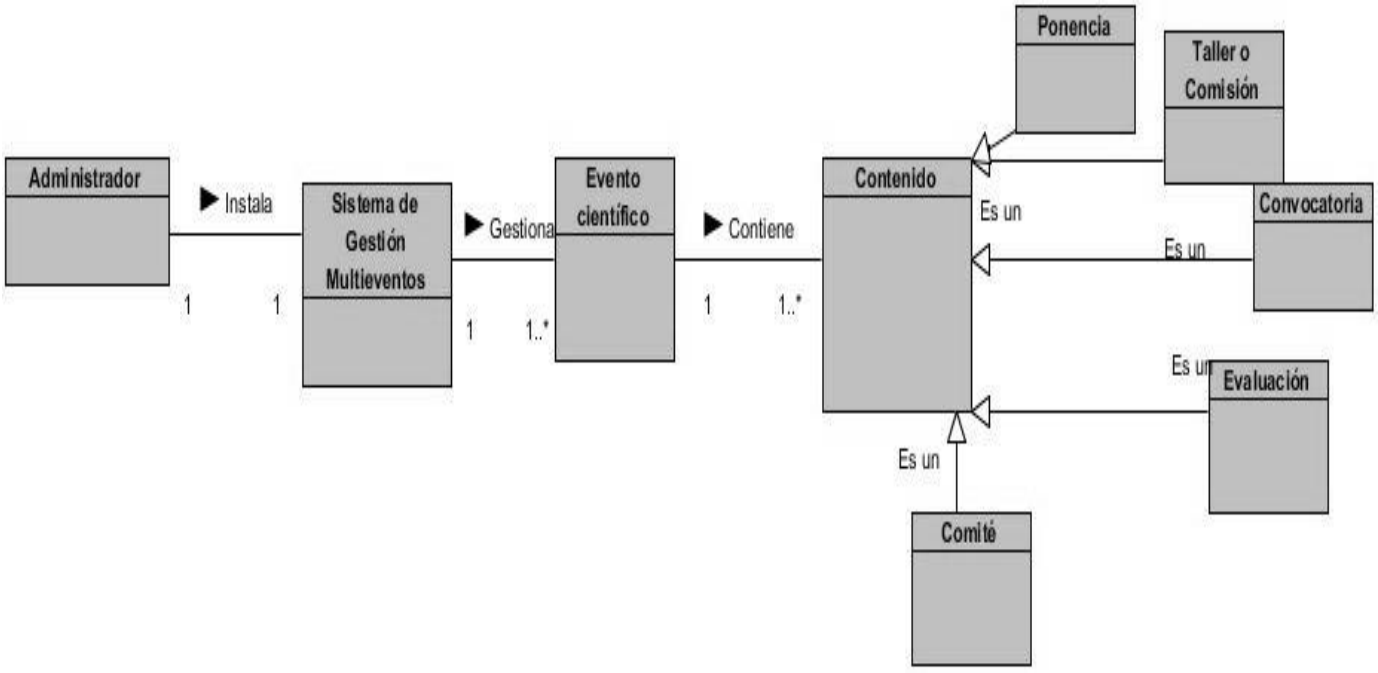



Figura 4. Modelo conceptual.

## 2.3 Técnicas para la captura de requisitos

Para la obtención de los requerimientos, se ha considerado como la opción más adecuada, la realización de reuniones periódicas con el cliente (en este caso, el líder del proyecto SGM) para ir definiendo las necesidades a cubrir.

## 2.4 Requisitos funcionales

El análisis de requerimientos consiste en la determinación de una serie de condiciones a satisfacer por el software que se debe desarrollar. En este apartado se hará una descripción del proyecto de carácter general, con sus funcionalidades principales y los usuarios que la conformarán.(Miguel Martín Romo 2004)

En el siguiente capítulo se especificarán cada una de las funcionalidades requeridas por el cliente. A continuación, se definen los requisitos funcionales para la realización del perfil de instalación. Consulte la especificación de requisitos en el Anexo # 1.

- RF1: Mostrar pantalla de bienvenida
- RF2: Seleccionar idioma
- RF3: Verificar requisitos
- RF4: Configurar base de datos
- RF5: Instalación del sistema
- RF6: Configurar traducciones
- RF7: Configurar sitio
- RF8: Configuración de dominios
- RF9: Adicionar dominio
- RF10: Seleccionar tema del dominio
- RF11: Mostrar progreso de la importación de la base de datos
- RF12: Importar el SQL del portal Multieventos
- Gestionar tipo de contenido Actividad del evento
    - RF13: Crear tipo de contenido Actividad del evento
    - RF14: Modificar tipo de contenido Actividad del evento
    - RF15: Eliminar tipo de contenido Actividad del evento

- Gestionar tipo de contenido Carousel
  - RF16: Crear tipo de contenido Carousel
  - RF17: Modificar tipo de contenido Carousel
  - RF18: Eliminar tipo de contenido Carousel
- Gestionar tipo de contenido Comité
  - RF19: Crear tipo de contenido Comité
  - RF20: Modificar tipo de contenido Comité
  - RF21: Eliminar tipo de contenido Comité
- Gestionar tipo de contenido Convocatoria
  - RF22: Crear tipo de contenido Convocatoria
  - RF23: Modificar tipo de contenido Convocatoria
  - RF24: Eliminar tipo de contenido Convocatoria
- Gestionar tipo de contenido Evaluaciones
  - RF25: Crear tipo de contenido Evaluaciones
  - RF26: Modificar tipo de contenido Evaluaciones
  - RF27: Eliminar tipo de contenido Evaluaciones
- Gestionar tipo de contenido Ponencia
  - RF28: Crear tipo de contenido Ponencia
  - RF29: Modificar tipo de contenido Ponencia
  - RF30: Eliminar tipo de contenido Ponencia
- Gestionar tipo de contenido Talleres o comisiones
  - RF31: Crear tipo de contenido Talleres o comisiones
  - RF32: Modificar tipo de contenido Talleres o comisiones

- RF33: Eliminar tipo de contenido Talleres o comisiones

## 2.5 Requisitos no funcionales

Después de analizar las funcionalidades que el perfil de instalación debe cumplir, es necesario analizar las propiedades o características que el mismo debe tener para un mejor funcionamiento, a esto se le conoce como requisitos no funcionales. Entre los requisitos no funcionales necesarios para el perfil de instalación se encuentran los siguientes:

- **Requisitos de usabilidad:** el perfil de instalación puede ser usado fácilmente por cualquier usuario ya que la interfaz presenta un diseño intuitivo y legible que permite en todo momento conocer el estado de ejecución de la instalación del sistema.
- **Requisitos para las restricciones del diseño:** el perfil de instalación tiene que ser usado con Drupal en su versión 7, como Sistema Gestor de Contenidos (CMS). El perfil tiene que ser desarrollado con PHP como lenguaje de programación del lado del servidor, en su versión: 5.2.5 o superior (5.6 recomendado). Es necesario utilizar PostgreSQL como tipo de base de datos.
- **Confiabilidad:** para que el perfil de instalación se considere confiable, debe cumplir con las normas de seguridad y buenas prácticas de programación establecidas por el equipo de desarrollo del sitio oficial de la comunidad de Drupal.
- **Ayuda y documentación en línea:** se debe incluir una manual de usuario, para satisfacer las necesidades del usuario en cuanto a funcionalidad, comprensión e información para un mejor uso del perfil de instalación.
- **Portabilidad:** el sistema debe ser capaz de ejecutarse en diferentes distribuciones Linux sin sufrir cambios en su estructura.
- **Interfaz:** el diseño de la interfaz visual debe ser minimalista, sin uso excesivo de animaciones e imágenes de alta calidad o tamaño que perjudiquen el tiempo de respuesta de las peticiones que se realicen a la aplicación. La interfaz debe ser diseñada de forma tal que permita una sencilla navegación y el fácil entendimiento de las funcionalidades que brinda además de poseer colores refrescantes para una mejor interacción entre el usuario y la aplicación.

## 2.6 Historias de Usuarios (HU)

Las historias de usuario se utilizan para especificar las funcionalidades del software, las mismas son escritas en un leguaje natural y con palabras concisas para no exceder su tamaño en unas pocas líneas de texto. Además de que son una guía para la construcción posterior de las pruebas comprobando de esta manera la correcta implementación de las historias de usuarios.

A continuación, será mostrada la HU perteneciente al requisito funcional Configurar base de datos. Para ver todas las Historias de usuario consultar ANEXO #2.

Tabla 3.HU Configurar base de datos**.**

| | |
|---|---|
| **Número:** 4 | **Nombre del requisito:** Configurar base de datos |
| **Usuario:** Administrador | **Iteración Asignada:** 1 |
| **Prioridad:** Media | **Tiempo Estimado:** 3 semanas |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 2 semanas |

**Descripción:**

**1- Objetivo:**

Permitir seleccionar la base de datos deseada.

**2- Acciones para lograr el objetivo (precondiciones y datos):**

-Verificar si existen los requisitos necesarios para continuar con la instalación.

**3- Comportamientos válidos y no válidos (flujo central y alternos):**

Los siguientes campos son obligatorios:

-Tipo de base de datos:

- PostgreSQL

-Nombre de la base de datos: Admite caracteres de la (A/Z). El nombre de la base de datos debe existir en el servidor antes de instalar Drupal.

-Nombre del usuario de la base de datos: Admite caracteres de la (A/Z).

**4- Flujo de la acción a realizar:**

-El instalador ofrece la opción Configurar base de datos automáticamente después de haber realizado con éxito el paso anterior.

-El sistema muestra un campo de selección con las opciones avanzadas las cuales inicialmente están predefinidas.

-Si los datos están incompletos o incorrectos se señalarán los campos en cuestión mostrándole un mensaje de error al usuario.

-Si el usuario inserta correctamente los datos necesarios debe seleccionar el botón Guardar y continuar.

**Observaciones:** HU correspondiente al requisito funcional 3

**Prototipo de interfaz:**


## 2.7 Diagrama de clases del análisis

Una clase de análisis representa una abstracción de una o varias clases, ajustando las mismas a uno de los tres estereotipos existentes sobre las clases utilizados por el modelo de dominio: de interfaz, de control o de entidad.

La clase Interfaz se utiliza generalmente para modelar la interacción entre el sistema y los actores, la clase de Control es utilizada habitualmente para representar coordinación, secuenciación, transacciones y son

las encargadas de manejar y coordinar las acciones y los flujos de control principal. Por su parte la clase Entidad es usada para modelar la información que tiene una vida larga y a veces es persistente, asimismo muestran una estructura de datos lógica y contribuyen a comprender de qué información depende el sistema. (Jacobson Booch 2014)

A continuación, se presenta el diagrama de clase del análisis perteneciente al requisito funcional: Configurar base de datos. Los diagramas de clase del análisis correspondiente a los restantes requisitos funcionales se muestran en el Anexo #3.

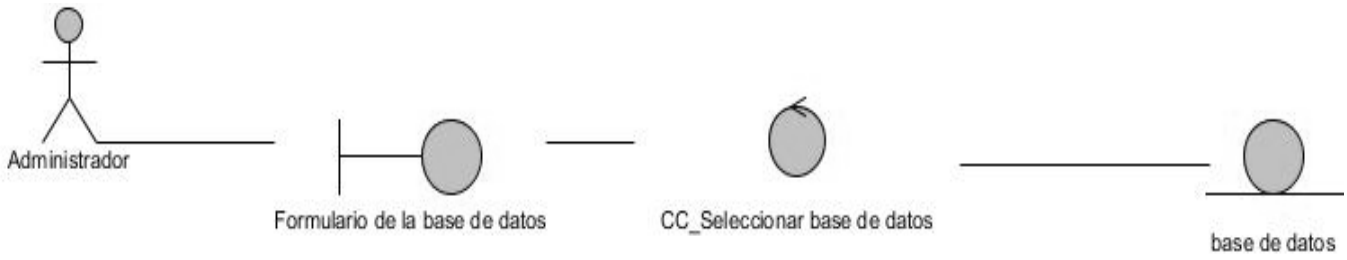



Figura 5. Diagrama de clase del análisis: Configurar base de datos.

## 2.8 Diagramas de colaboración

Los diagramas de colaboración del análisis son utilizados fundamentalmente para modelar las interacciones entre los objetos en el análisis. Estos recuerdan los diagramas de clases pero contienen instancias y enlaces en lugar de clases y asociaciones, mostrando cómo interactúan los objetos secuencialmente o en paralelo enumerando los mensajes que se envían unos a otros. (Jacobson Booch 2014)

A continuación, se muestra el diagrama de colaboración correspondiente al requisito funcional: Configurar base de datos. Los diagramas de colaboración correspondiente a los restantes requisitos funcionales se muestran en el Anexo #4.

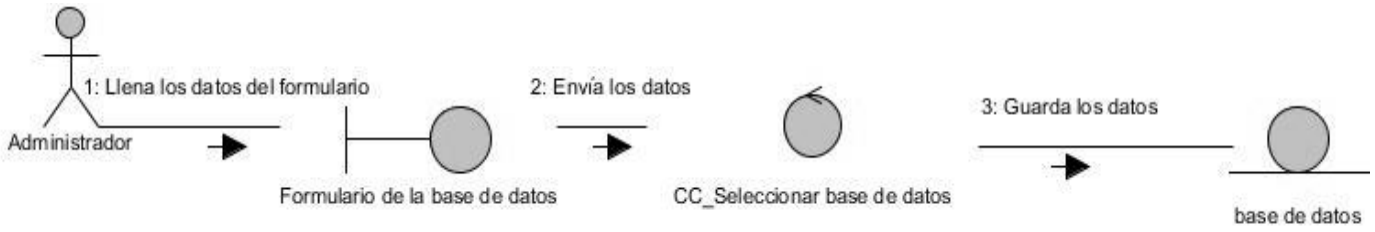



Figura 6. Diagrama de colaboración: Configurar base de datos.

## 2.9 Patrones de diseño

"*Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular*".(Aldibier Morales Morales 2010; García 2014; Adrián Ramos 2014)

### 2.9.1 Patrones GOF

Los patrones GOF (Gang Of Four) muestran soluciones para clases de problemas muy particulares en el diseño orientado a objetos, en Drupal se pueden identificar algunos como los siguientes:

- ✓ **Creacionales**: los patrones creacionales abstraen el proceso de creación de instancias y ocultan los detalles de cómo los objetos son creados o inicializados.
- ✓ **Estructurales**: los patrones estructurales se ocupan de cómo las clases y objetos se combinan para formar grandes estructuras y proporcionar nuevas funcionalidades.
- ✓ **Comportamiento**: los patrones de comportamiento están relacionados con los algoritmos y la asignación de responsabilidades entre los objetos. Son utilizados para organizar, manejar y combinar comportamientos.

**En la propuesta de solución se usan los siguientes patrones de diseño:**

- ✓ Bridge (Puente): es una técnica usada en programación para desacoplar una abstracción de su implementación, de manera que ambas puedan ser modificadas independientemente sin necesidad de alterar por ello la otra. En Drupal la capa de abstracción de la base de datos es implementada de una manera similar, imitando el patrón Bridge, no realizando llamadas directas a la base de datos,

sino que se hacen a través de funciones genéricas definidas por la capa de abstracción, funcionando ésta como puente. Estas funciones no varían, independientemente del gestor de bases de datos utilizado.

Ejemplo de la capa de abstracción de la base de datos a través de la función de la Database API de Drupal db_update. Esto garantiza que, sin importar el gestor de base de datos en cuestión, la consulta se ejecutará de la misma forma, evitando así tener que definir diferentes métodos para cada gestor.

```
db_update('users')->fields(array(
'name' => $name,
'pass' => user_hash_password($pass),
'mail' => $mail,
'data' => '',
'picture' => 0,
))
->condition('uid', 1)
->execute();
```

- ✓ Chain of Responsibility (Cadena de responsabilidades): patrón de comportamiento que evita acoplar el emisor de una petición a su receptor dando a más de un objeto la posibilidad de responder a una petición. Para ello, se encadenan los receptores y pasa la petición a través de la cadena hasta que es procesada por algún objeto.

Este patrón es utilizado a menudo en el contexto de las interfaces gráficas de usuario donde un objeto puede contener varios objetos. Según si el ambiente de ventanas genera eventos, los objetos los manejan o los pasan. En el sistema de menús de Drupal se puede identificar el patrón de cadena de responsabilidades.

Cuando el usuario visita una página web, el sistema debe determinar, a partir de la URL, qué módulo es el encargado de su gestión y presentación, siguiendo así un flujo de trabajo en cadena comprobando en todos los módulos activos del sitio a qué módulo pertenece cada solicitud de página, buscando las

implementaciones de hook_menu(). Una vez encontrado el módulo, este le facilitará al sistema una función de retorno que es la encargada de generar el contenido de la página.

Se evidencia en la function sgm_domain_settings_form_validate ($form, &$form_state) {...}

Al llamar a la función de validación del formulario de dominios se llama a su vez a la función de validación del módulo dominio en sí. Esta petición es enviada y se hace una búsqueda en cadena a través de todos los módulos del sitio hasta que se encuentre está implementación y la petición se atendida.

## 2.10 Patrones Arquitectónicos

El CMS Drupal hace uso del patrón arquitectónico Modelo-Vista-Controlador (MVC) mediante los templates (vistas), los hooks (controlador) y la base de datos (modelo). Los templates son los primeros en interactuar con el usuario y son los que se encargan de recibir los datos, para que luego los hooks trabajen en la validación, la base de datos procesa y gestiona la petición para luego dar un resultado al usuario por medio de la vista.

Para una mejor comprensión de cómo trabaja el patrón MVC en Drupal es necesario conocer sobre la arquitectura de las capas, ellas hacen que Drupal sea un CMS flexible y esto posibilite mantener los componentes organizados. A continuación, se muestra una gráfica con las 5 capas que posee Drupal. La presente propuesta de solución presenta una arquitectura en capas (Figura Arquitectura de Drupal), las cuales son descritas a continuación:

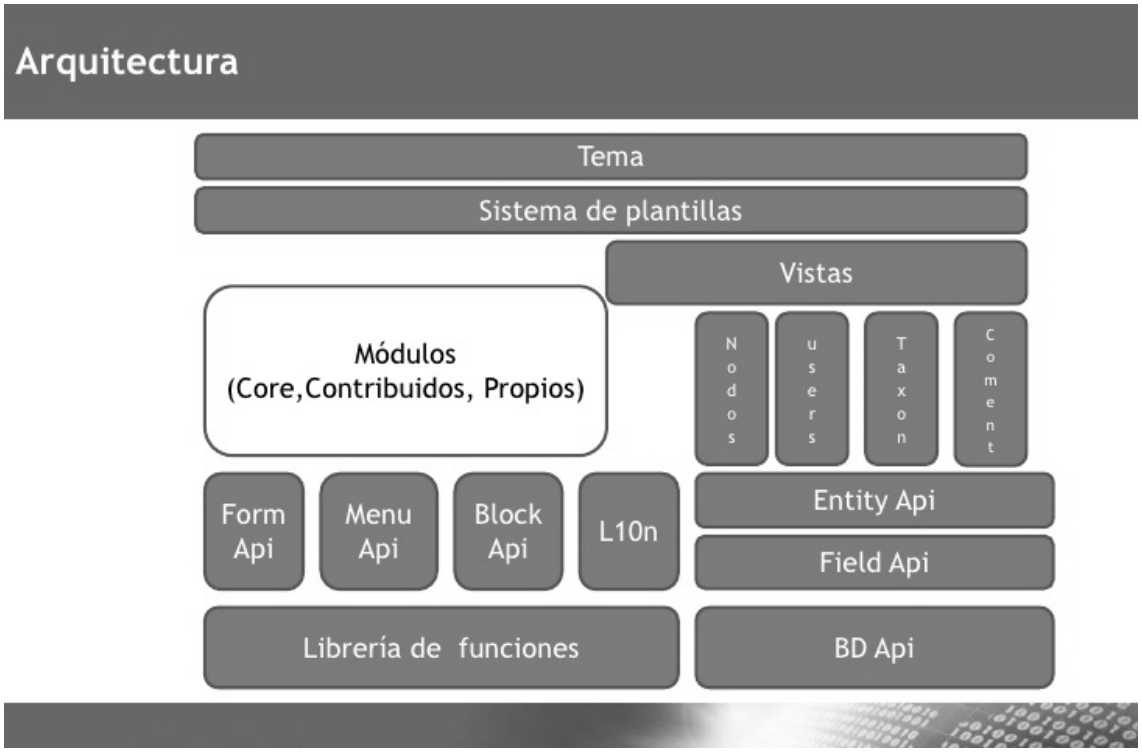



Fig. 7. Arquitectura de Drupal. Estructura en 5 capas. (Aldibier Morales Morales 2009)

**Temas (*theme*):** esta capa establece la apariencia gráfica que se le muestra al usuario. Esta separación entre información y los estilos permite cambiar la apariencia del portal web sin necesidad de modificar los contenidos y funcionalidades implementadas. Los temas también pueden ser regulados sobre la base de permisos de usuario.

**Vistas (*views*):** es la capa encargada de mostrar en los temas los cambios realizados a través de los módulos. Se complementan con las entidades y campos ampliando las posibilidades de presentación de contenidos dinámicos.

**Entidades (*entities*):** representa las entidades, que engloban los nodos, los usuarios, las taxonomías y los comentarios. Esta nueva estructura permite que sea posible añadirle campos a todo aquello que sea una entidad.

**Módulos (*modules*):** engloba los elementos, *plugins*, que operan sobre los nodos, usuarios, menús y administración, fusionándose al núcleo de Drupal para la creación de nuevas funcionalidades al sistema. Permite incrementar sus capacidades o adaptarlas a las necesidades de cada aplicación

**Base de Datos (*database*):** esta capa es la encargada de gestionar el acceso a la información almacenada referente al funcionamiento del sistema y a los contenidos que serán mostrados a través del tema activo. (José Antoni Dorado Cerón 2015)

## 2.11 Diagrama de paquetes del diseño

Para una mejor comprensión del sistema a desarrollar, es necesario tener en cuenta el diagrama de paquetes del CMS Drupal. A continuación, se describen los principales paquetes que lo componen:

***"Themes"*:** lugar donde se encuentran ubicadas las plantillas que se utilizan como tema en la interfaz de Drupal, cuando se necesita incluir un nuevo diseño se copia la plantilla dentro de esta carpeta.

***"Modules"*:** abarca todos los módulos para el funcionamiento del CMS, cuando se requiere un nuevo módulo solo debe copiarse en esa carpeta.

***"Includes"*:** contiene ficheros indispensables para el correcto funcionamiento del sistema, entre los que se encuentran los de conexión a la base de datos.

***"Scripts"*:** agrupa el conjunto de ficheros que posibilitan la visualización de los datos, destacándose los CSS y JavaScript.(Aldibier Morales Morales 2009)

Un diagrama de clase del diseño contiene una mayor cantidad de detalles que el diagrama de análisis entre los que se encuentran: clases, atributos, operaciones, subsistemas y relaciones.(Jacobson Booch 2014)

A continuación, se muestra el diagrama de clase del diseño correspondiente al requisito funcional: Mostrar pantalla de bienvenida.

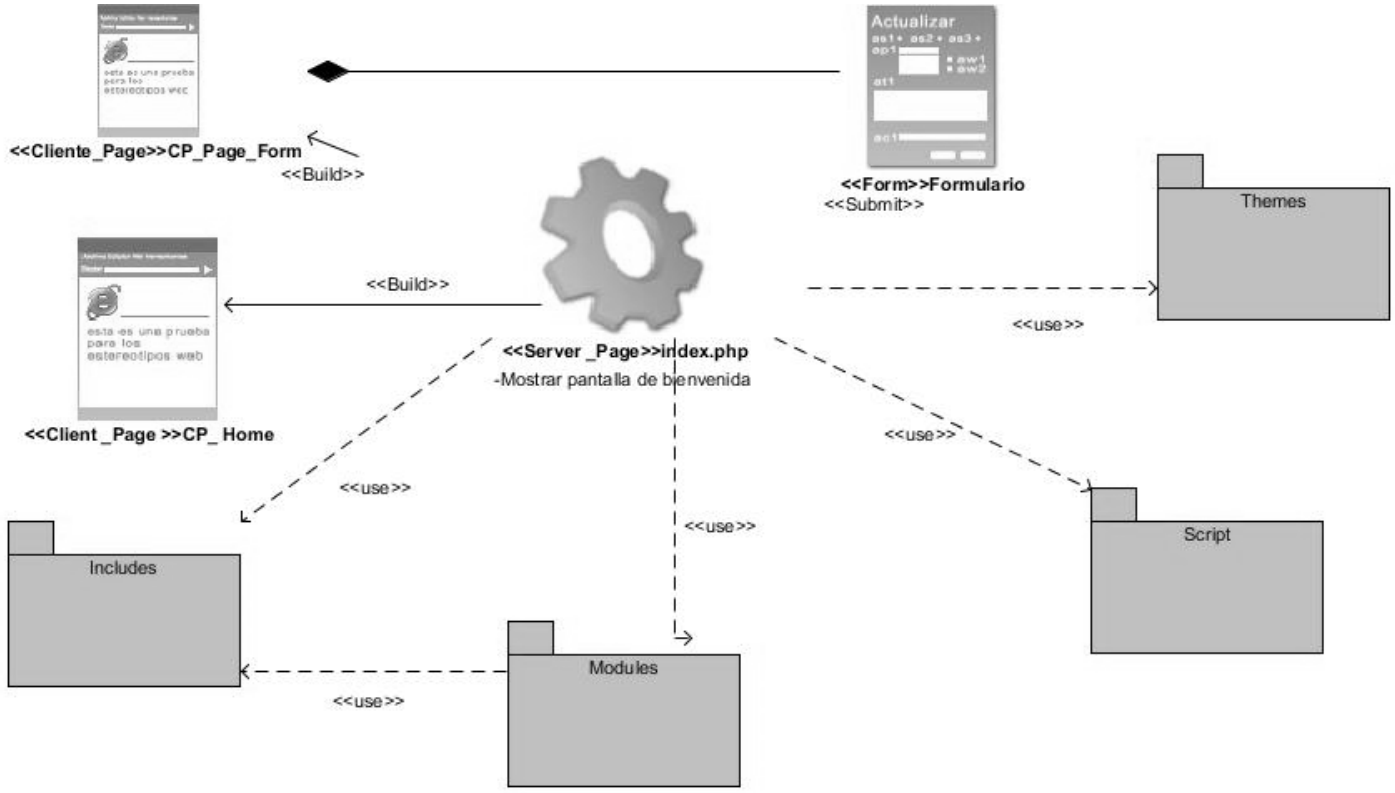



Figura 7. Diagrama de clases del diseño mostrar pantalla de bienvenida.

## 2.12 Diagramas de secuencia

El diagrama de secuencia muestra cómo se pasa de un objeto a otro a medida que se ejecuta el requisito funcional relacionándose a través de mensajes. A menudo los desarrolladores utilizan textos para explicar cómo interactúan los objetos de diseño para llevar a cabo el flujo de eventos(Jacobson Booch 2014).

A continuación, se muestra el diagrama de secuencia correspondiente al requisito funcional: Configurar base de datos. Los restantes diagramas se muestran en el Anexo #5.

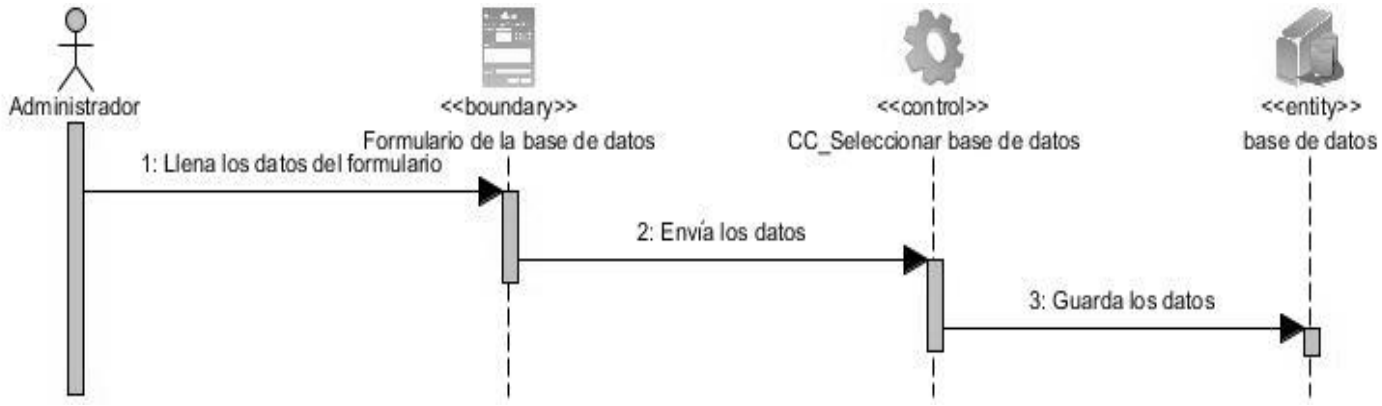



Figura 8. Diagrama de secuencia Configurar base de datos.

## 2.13 Modelo de datos

El diagrama entidad-relación es uno de los modelos más usados para diseñar bases de datos, este modelo se encuentra basado en dos conceptos fundamentales: entidades, que representan objetos sobre los cuales se desea guardar información y las relaciones, que constituyen las relaciones entre las entidades.

A continuación, se presenta el modelo de datos que contiene las entidades que serán utilizadas por las funcionalidades a desarrollar y las relaciones entre ellas, las mismas representan las tablas en la base de datos**.** La descripción de las tablas se encuentra en el Anexo#6.

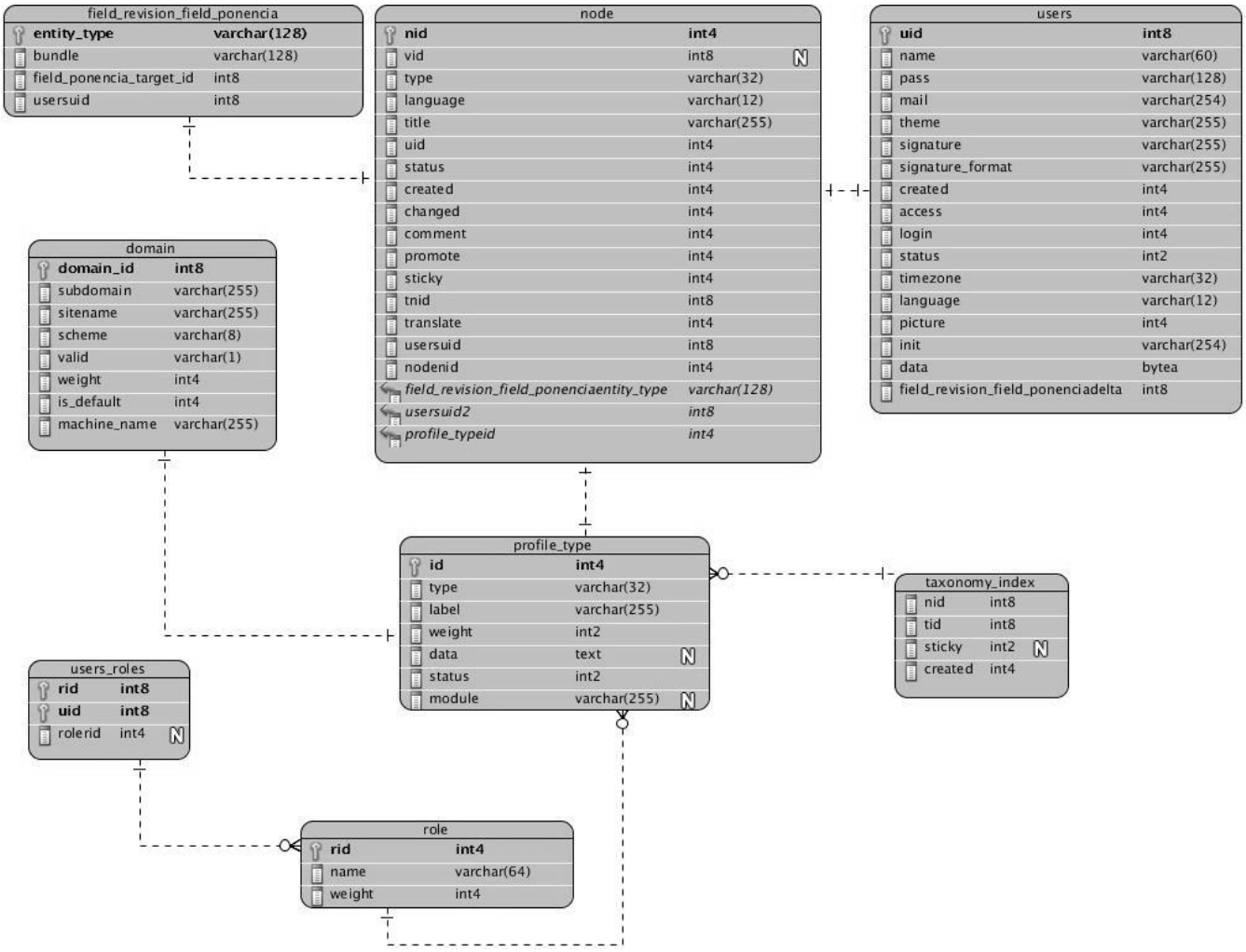



Figura 9. Modelo de datos.

## 2.14 Diagrama de despliegue

El diagrama de despliegue es utilizado para mostrar los nodos y conexiones del modelo de despliegue, así como la asignación de los objetos a los nodos (Jacobson, 2000).

El diagrama que se presenta a continuación representa la distribución física del sistema a través de nodos, está compuesto por una PC cliente que deberá tener instalado un navegador web, donde la comunicación entre ella y los servidores se llevará a cabo a través del Protocolo Seguro de Transferencia de Hipertexto (HTTPS por sus siglas en inglés).

El sistema contará además con un servidor web (apache), gestor de base de datos (postgresql) y un servidor Nginx, estos servidores se comunicarán vía TCP-IP (familia de protocolos de Internet compuesta fundamentalmente por el Protocolo de Control de Transmisión, TCP por sus siglas en inglés, y el Protocolo de Internet, IP por sus siglas en inglés).

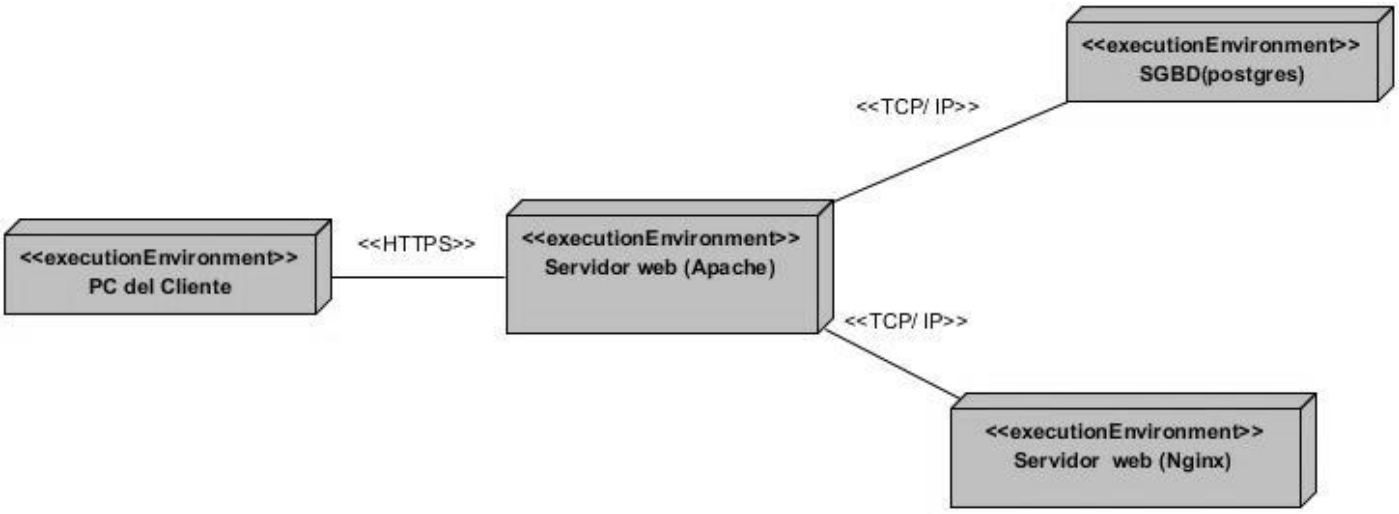



Figura 10. Diagrama de despliegue.

## 2.15 Conclusiones

Después de haber concluido este capítulo quedan demostrados los elementos necesarios para llegar a brindar una propuesta de solución para la investigación en cuestión, concluyéndose que:

- La elaboración del modelo conceptual contribuyó a esclarecer las relaciones entre los principales conceptos del dominio del problema.
- La obtención de las especificaciones de requisitos del software posibilitó un mejor entendimiento y desglose de las acciones a realizar por el instalador.
- El uso de los patrones de diseño y patrones arquitectónicos posibilitó mantener los componentes organizados, logrando así una mejor interacción entre los datos.

- El desarrollo del flujo de análisis y diseño correspondiente a la metodología seleccionada, identificó los elementos que componen tanto la arquitectura como el diseño de la herramienta a implementar y la generación de los artefactos correspondientes para la posterior implementación.

# *Capítulo 3*

## *Implementación y prueba del perfil de instalación*

## Introducción

Cuando se desarrollan aplicaciones en equipo o en comunidad, es importante que el código generado sea fácil de leer y modificar independientemente de quién haya sido el autor. El primer paso para lograr esto será seguir una serie de normas o pautas para formatear el código de una forma común a todos los desarrolladores. (Fran Gil Rodríguez 2011a)

A continuación, se definen los estándares de codificación utilizados por Drupal, definiendo así un estilo de programación homogéneo, el cual permitirá un total entendimiento del código desarrollado. Además de realizar diferentes pruebas de software que garantizarán un producto final con la calidad máxima requerida.

## 3.1 Estándares de codificación

### Indentación:

En Drupal se debe indentar con 2 espacios, nunca con tabuladores. Además, no se debe dejar espacios en blanco al final de cada línea. En el siguiente ejemplo se muestra un fragmento de código con las indentaciones realizadas, de 2 espacios cada una, y los saltos de línea o Enter al final de cada línea (sin dejar espacios).

Ejemplo:

```
if (!function_exists("system_form_install_select_profile_form_alter")) {
function system_form_install_select_profile_form_alter(&$form, $form_state) {
foreach ($form['profile'] as $key => $element) {
$form['profile'][$key]['#value'] = 'sgm';
}
}
}
```

**Etiquetas de apertura y cierre de PHP**: cuando se esté escribiendo en PHP, siempre se deben utilizar las etiquetas <?php y ?>, y en ningún caso la versión corta < y ?>. En general se omite la etiqueta de cierre de PHP (?>) al final de los archivos.module y.inc.

Esta convención evita que se puedan quedar olvida un espacio no deseados al final del archivo (después de la etiqueta de cierre ?>), que serían identificados como salida HTML y podrían provocar un error muy típico, " Cannot modify header information - headers already sent by". Por tanto, la etiqueta de cierre final del archivo (?>) es opcional en Drupal.

**Operadores:** los operadores binarios, que se utilizan entre dos valores, deben separarse de estos valores, a ambos lados del operador, por un espacio. Por ejemplo, $numero = 3, en lugar de $numero=3. Esto se aplica a operadores como +, -, *, /, =, ==, !=, >, <, . (concatenación de cadenas), .=, +=, -=, etc. Los operadores unarios como ++, -- no deben tener separación. Por ejemplo, $numero++.

Ejemplo:

$tasks = array_slice($old_tasks, 0, 1) + $new_task + array_slice($old_tasks, 1);

**Uso de comillas:** se pueden usar tanto las comillas simples ('cadena') como las comillas dobles ("cadena") para delimitar las cadenas de caracteres. Las comillas dobles son necesarias si se desean incluir variables dentro de las cadenas de texto.

Ejemplo:

fputs($f, '$conf["drupal_private_key"]' . '=' . """ . $key . """ . ';' . "\n");

**Uso de punto y coma (;) en código PHP:** aunque PHP permite escribir líneas de código individuales sin el terminador de línea (;), como por ejemplo <?php print $title?>. En Drupal es siempre obligatorio: <?php print $title; ?>. - Correcto: <?php print $title; ?> - Incorrecto: <?php print $title ?>

Correcto: <?php print $title; ?>

Incorrecto: <?php print $title ?>

Ejemplo:

$success = TRUE;

$query = '';

$new_line = TRUE;

**Estructuras de control:** con respecto a las estructuras de control, hay que tener en cuenta las siguientes normas:

- Debe haber un espacio entre el comando que define la estructura (if, while, for, etc.) y el paréntesis de apertura. Esto es así para no confundir las estructuras de control con la nomenclatura de las funciones, como se evidencia más adelante.
- La llave de apertura {, se situará en la misma línea que la definición de la estructura, separada por un espacio.
- Se recomienda usar siempre las llaves {} aún en los casos en que no sea obligatorio su uso (una sola "línea" de código dentro de la estructura de control).
- Las estructuras else y elseif se escribirán en la línea siguiente al cierre de la sentencia anterior.

Ejemplo:

```
if (!function_exists("system_form_install_configure_form_alter")) {
function system_form_install_configure_form_alter(&$form, $form_state) {
$form['site_information']['site_name']['#default_value'] = 'Sistema de Gestión Multieventos';
}
}
```

**Funciones:** los nombres de las funciones deben estar escritos en minúsculas y las palabras separadas por guión bajo. Además, se debe incluir siempre como prefijo el nombre del módulo, tema, etc., para evitar así duplicidad de funciones. En su declaración, después del nombre de la función, el paréntesis de inicio de los argumentos debe ir sin espacio. Cada argumento debe ir separado por un espacio, después de la coma del argumento anterior.

Ejemplo:

```
function sgm_install_disable_outdated($key) {}
```

En la llamada a la función se aplican las mismas reglas anteriores con respecto a los parámetros, como se muestra en el siguiente ejemplo:

$mistake = domain_validate($form_state['values']['domain_text']);

**Arrays:** los valores dentro de un array (o matriz) se deben separar por un espacio (después de la coma que los separa). El operador => debe separarse por un espacio a ambos lados.

Cuando la línea de declaración del array supera los 80 caracteres, cada elemento se debe escribir en una única línea, indentándolo una vez (2 espacios). En este último caso, la coma de separación del último elemento también se escribirá, aunque no existan más elementos. De esta forma se evitan errores al añadir nuevos elementos al vector.

Ejemplo:

$form['actions']['submit'] = array(

'#type' => 'submit',

'#value' => st("Guardar y finalizar"),

'#weight' => 10,

);

**Constantes:** los nombres de las constantes deben escribirse en mayúsculas, con guiones bajos para separar palabras. Al igual que ocurre con las funciones, los nombres de las constantes deben tener como prefijo el nombre del módulo (o tema) en el que se utilizan, para evitar errores de duplicidad de constantes. Este prefijo también se escribirá en mayúsculas.

Ejemplo:

'run' => 'INSTALL_TASK_RUN_IF_NOT_COMPLETED',

**Comentarios:** para realizar comentarios de documentación se utiliza la siguiente sintaxis: /** * Comentarios para documentación */ (Fran Gil Rodríguez 2011a)

## 3.2 Diagrama de componentes

El diagrama de componentes muestra los tipos de componentes del sistema; una configuración particular de la aplicación puede tener más de una copia de un componente. Un diagrama de componentes muestra dependencias entre los componentes.

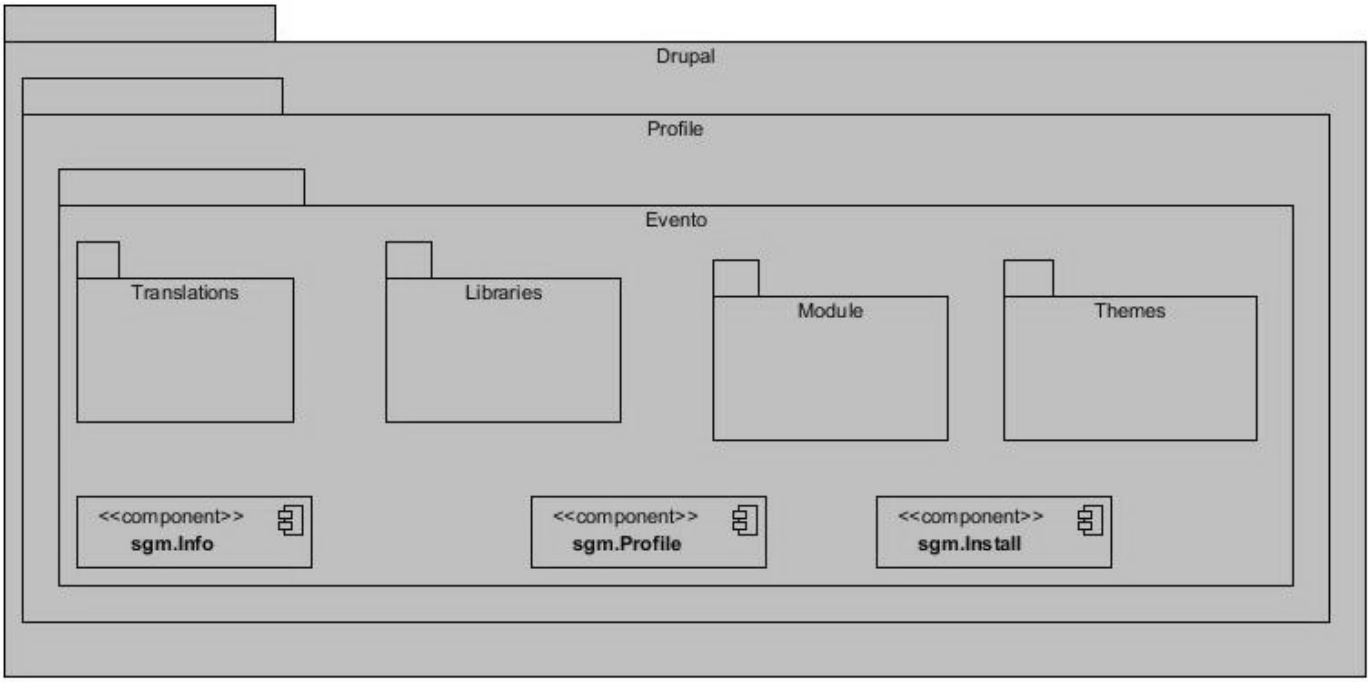



Figura 11. Diagrama de componentes.

En la figura 11 se muestra el diagrama de componentes del perfil de instalación utilizando Drupal en su versión 7, formado por los elementos que lo integran:

**sgm.info:** es el archivo donde se define el perfil de instalación. El contenido es similar al archivo.info de módulos y temas, además de contener información básica sobre el perfil (nombre, descripción, versión de Drupal, etc.), así como dependencias con otros módulos.

**sgm.profile:** es el archivo donde se implementan las tareas que se llevarán a cabo durante la instalación. Es equivalente al archivo. module de los módulos y en él se pueden usar la mayoría de funciones disponibles de la API de Drupal.

**sgm.install:** permite implementar **hook_install** () para insertar contenido en la base de datos durante la instalación. Permite activar módulos propios del perfil de instalación que estarán disponibles en el sitio después de la instalación.

**translations:** en esta carpeta se añaden las traducciones del núcleo, necesarias para que el proceso de instalación se realice en otro idioma (por ejemplo, en español).

**libraries:** es una carpeta donde se ubican las librerías a usar, que son necesarias para extender las funcionalidades en el sitio.

**modules:** en esta carpeta se añaden los módulos que estarán disponibles durante la instalación. Generalmente sólo se usa esta carpeta para añadir módulos creados específicamente para la distribución. El resto de módulos requeridos serán registrados como dependencias, de forma que se soliciten durante la instalación.

**themes:** en esta carpeta se pueden añadir temas que estarán disponibles durante la instalación. Generalmente sólo se usa esta carpeta para añadir temas creados específicamente para la distribución.

## 3.3 Pruebas de Software

*"Pruebas de Software: es la ejecución del código usando combinaciones de entradas, en un determinado estado, para revelar defectos. Proceso de ejecutar un programa con el fin de encontrar errores."*(Ivette Carolina Martínez 2016)

Actividad en la cual un sistema o uno de sus componentes se ejecutan en circunstancias previamente especificadas, los resultados se observan y registran para posteriormente realizar una evaluación de algún aspecto. Las pruebas se aplican con diferentes objetivos y generalmente en distintos escenarios de trabajo. A continuación, se describen las pruebas aplicadas al instalador.

### 3.3.1Tipos de prueba

Pruebas funcionales: están centradas en comprobar que las funcionalidades descritas en el documento de requisitos del sistema se cumplen con la implementación realizada. Los analistas enfocan su atención a

las respuestas del sistema de acuerdo a los datos de entrada y sus resultados en los datos de salida, los cuales se definen generalmente en los casos de prueba que se crean antes del inicio de las pruebas.

### 3.3.2 Métodos de prueba

**Pruebas de caja blanca o estructural**: se basan en un minucioso análisis de los detalles procedurales del código a evaluar, por lo que es necesario conocer la lógica del programa. Su uso posibilita la obtención de casos de prueba que garantizan, al menos una vez, que sean ejecutados todos los caminos independientes de cada módulo. Posibilita ejercitar todas las decisiones lógicas en sus vertientes verdaderas y falsas. Permite, además, la ejecución de cada bucle con sus límites operacionales.

**Pruebas de caja negra:** *"Las pruebas de caja negra son las que se aplican a la interfaz del software. Una prueba de este tipo examina algún aspecto funcional de un sistema que tiene poca relación con la estructura lógica interna del software."*(Ivette Carolina Martínez 2016)

Pruebas que se llevan a cabo sobre la interfaz del software. El objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene (no se ve el código).

### 3.3.3 Técnicas

**Análisis de valores límites:** prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

**Grafos Causa-Efecto:** permite validar complejos conjuntos de acciones y condiciones.

**Técnica de la Partición de Equivalencia**: se utilizó esta técnica debido a que divide el dominio de entrada de un programa en un número finito de variables de equivalencia. Define dos tipos de variables de equivalencia, las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real de dato.

### 3.3.4 Estrategia: Casos de pruebas

*"Conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo en particular"*. (Ivette Carolina Martínez 2016)

En los casos de prueba (CP) se incluyen la descripción de los principales escenarios, actores, posibles entradas, variables que intervienen en el proceso y flujo central donde se realiza el procedimiento. A continuación, se presenta el CP propuesto para el requisito funcional Configurar base de datos. Para consultar el resto de los CP remitirse al Anexo #7.

Tabla 4. CP configurar base de datos.

| Escenario | Descripción | Tipo de base de datos | Nombre de la base de datos | Nombre del usuario | Contraseña de la base de datos | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|---|---|
| **EC1.1**<br>Configurar base de datos | Permite la configuración de la base de datos mediante un formulario. | | | | | Brinda la posibilidad de introducir de manera obligatoria los siguientes datos: Tipo de base de datos*<br>Nombre de la Base de datos*<br>Nombre del usuario de la base de datos*<br>De forma opcional: contraseña de la base | Configurar base de datos |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | de datos.<br>También muestra un campo de selección con las opciones avanzadas.<br>Permite: guardar las configuraciones realizadas y continuar el proceso de instalación. | |
| **EC 1.2** Opción de Guardar y continuar | Introduce y/o selecciona los datos de la configuración de la base de datos y selecciona la opción guardar y continuar. | V | V | V | V | Valida los datos. Continúa el proceso de instalación. | Configurar base de datos/Guardar y continuar |
| **EC 1.4** Datos | Existe algún | I | V | V | N/A | Muestra un | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| obligatorios incompletos | campo obligatorio incompleto | V | I | V | N/A | mensaje de error informando que el campo es obligatorio.<br><br>Regresa al EC 1.1. | Configurar base de datos/Guardar y continuar |
| | | V | V | V | N/A | | |
| **EC 1.5** Datos incorrectos | Existen datos incorrectos. | I | V | | N/A | Muestra un mensaje de información. Regresa al EC 1.1. | Configurar base de datos/Guardar y continuar |
| | | V | I | | N/A | | |
| | | V | V | | N/A | | |

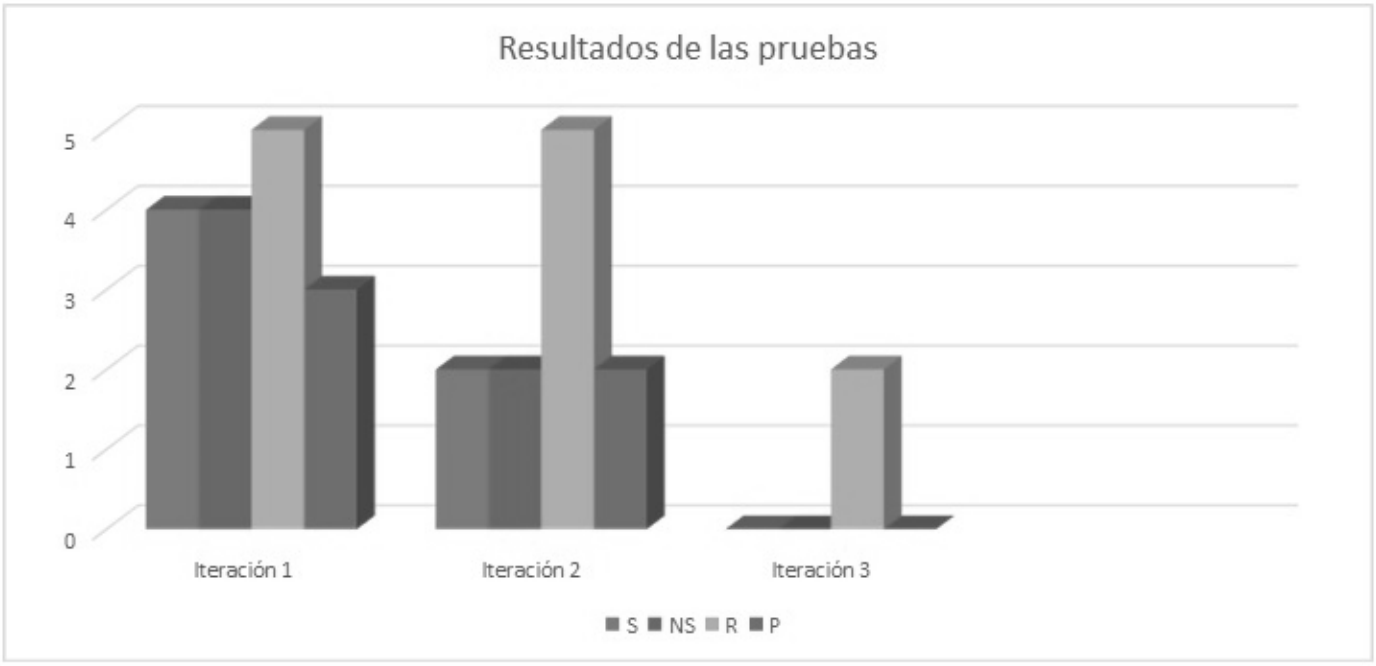
### 3.4 Resultados de las pruebas funcionales

Con el objetivo de verificar el cumplimiento de los requisitos funcionales establecidos para la presente investigación se hace uso del método de prueba caja negra, teniendo en cuenta la técnica de partición por equivalencia.

Además, se hace uso de los casos de prueba generados durante este flujo de trabajo con el fin de detectar la mayor cantidad de no conformidades posibles en las funcionalidades del componente realizándose tres iteraciones de prueba.

Luego de realizar las pruebas y analizar los resultados arrojados en todas las iteraciones, se corrigieron las deficiencias encontradas en cada una de ellas. Las deficiencias detectadas se definen como No Conformidades (NC).

Se clasifican en Significativas (S), No Significativas (NS), Pendientes (P) y Resueltas(R). Entiéndase por S aquellas NC que puedan afectar el funcionamiento del instalador, NS las NC orientadas al diseño u otro aspecto que no afecte el funcionamiento de la propuesta de solución, P las NC que no se resolvieron en la iteración encontrada y R todas aquellas que se han solucionado. El resultado obtenido se evidencia en el gráfico que se muestra a continuación, donde se puede observar la cantidad de NC de cada tipo detectadas en cada iteración.




Gráfica 1. Resultados de las pruebas.

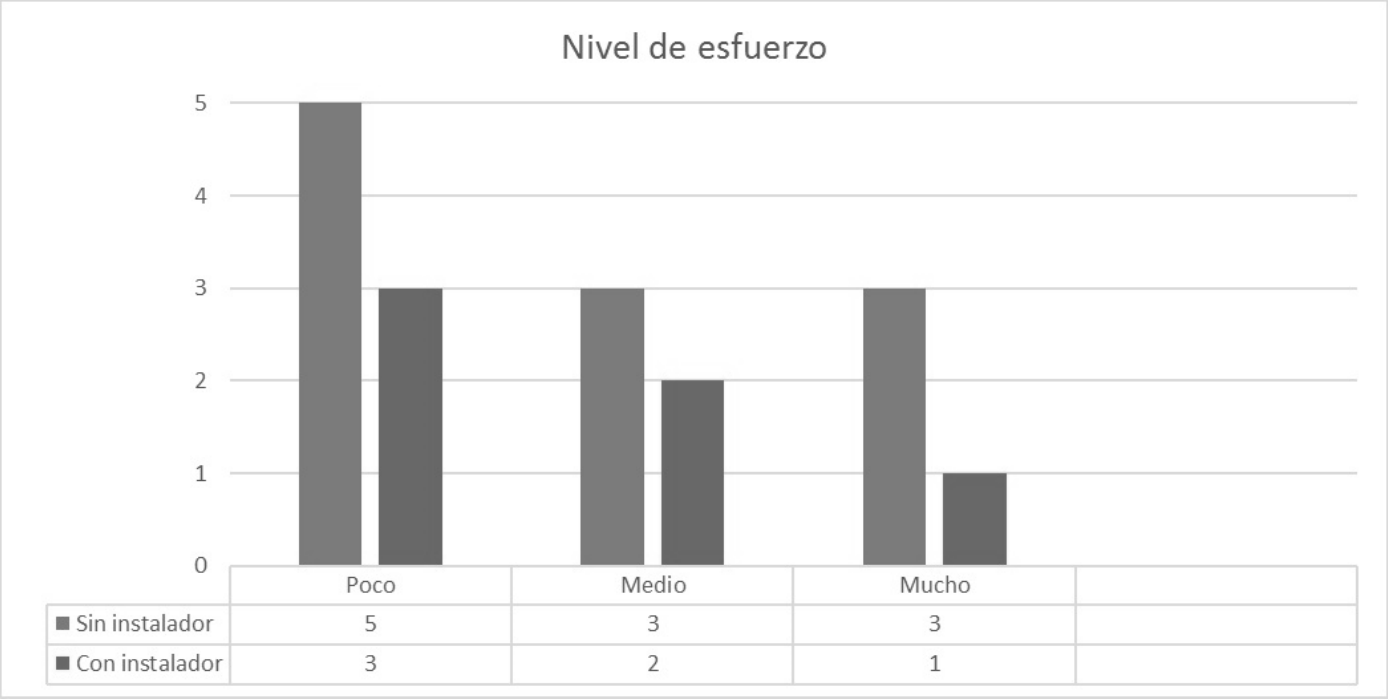
Como se demuestra en la gráfica anterior se detectaron 8 NC en la primera iteración y 4 NC en la segunda, ya en la tercera iteración quedaron todas las NC resueltas. La mitad de las NC son NS, lo que demuestra pocas deficiencias que pueden afectar el funcionamiento de la propuesta de solución.

**Nivel de esfuerzo empleado para el despliegue del SGM**

Con el objetivo de medir el nivel de esfuerzo empelado para el despliegue del SGM utilizando o no el instalador se realizó un muestreo intencional; su fundamento consiste en que el buen juicio posibilitará escoger los integrantes de la muestra, por lo que el investigador seleccionará explícitamente los elementos que son representativos o con posibilidades de brindar mayor información.

En este caso se utilizó una muestra de 5 personas pertenecientes al proyecto Multieventos. Se registraron los resultados teniendo en cuenta la escala de medición de Likert ordenada del 0 al 5, teniendo en cuenta que el esfuerzo estará estrechamente relacionado con el nivel de conocimiento (poco, medio o mucho) que posea la persona encargada de desplegar el sistema en cuento a:

- Habilidades con el gestor de base de datos PostgreSQL.
- Trabajo con archivos de configuración en el CMS Drupal en su versión 7.
- Dominio del SGM.




Gráfica 2. Nivel de esfuerzo empleado para el despliegue del SGM.

A partir de los resultados obtenidos en la gráfica se concluye que es necesario emplear menos esfuerzo en el proceso de despliegue del SGM con el uso de un instalador. Los beneficios obtenidos son los siguientes:

- Solucionada la necesidad de establecer un nuevo usuario y contraseña para el nuevo administrador del sitio.
- Se reducen las modificaciones manuales en el archivo settings.php relacionadas con especificar la base de datos.
- Se crea un nuevo dominio sin necesidad de completar todos los datos ya que algunos se consideran innecesarios al mismo tiempo que se le asigna un tema visual.
- Facilidad de configuración para usuarios con conocimientos básicos ya que las tareas de instalación se encargan de validar y notificar todo el proceso permitiendo una instalación exitosa.
- El sistema puede ser instalado remotamente por especialistas del centro FORTES ahorrando así recursos y tiempo.
- Se obtiene el sistema sin contenidos creados, sin usuarios adicionales excepto el administrador y sin dominios definidos por defecto garantizado así la reducción de datos innecesarios en la base de datos.

## 3.5 Pruebas no funcionales

Las pruebas no funcionales se realizan para verificar que el software desarrollado cumple con los requerimientos no funcionales establecidos por el cliente. Existen varios tipos de pruebas no funcionales, entre las más comunes están las pruebas de usabilidad, pruebas de rendimiento y pruebas de integración.

### 3.5.1 Pruebas de usabilidad

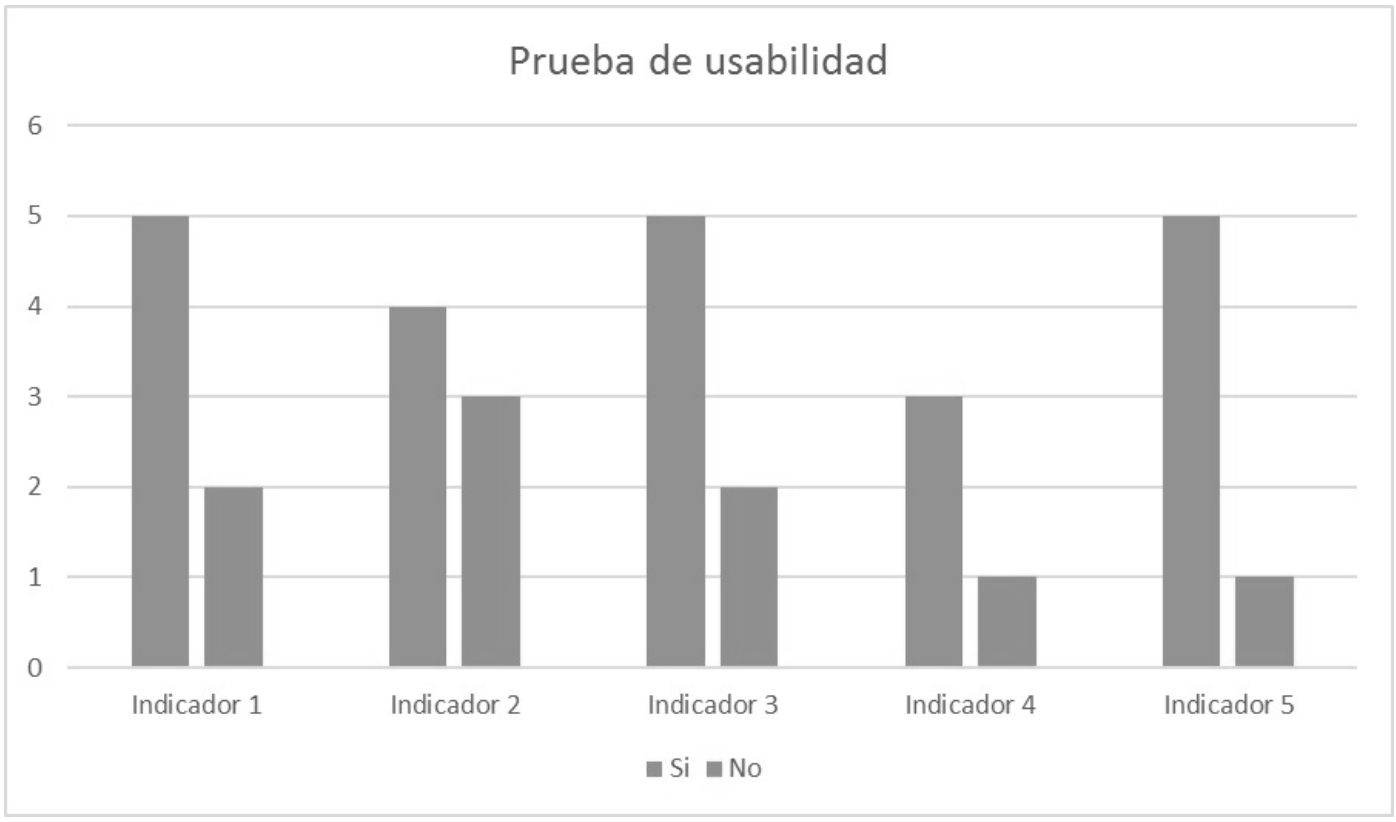
Las pruebas de usabilidad consisten en seleccionar a un grupo de usuarios de una aplicación y solicitarles que lleven a cabo las tareas para las cuales fue diseñada, en tanto el equipo de diseño, desarrollo y otros involucrados toman nota de la interacción, particularmente de los errores y dificultades con las que se encuentren los usuarios.

Para realizar estas pruebas se aplicó un test a los miembros del proyecto Multieventos, tomando como criterio para su selección que estén destinado a ser usuarios finales del sistema y su experiencia trabajando con el SGM. Ver test de usabilidad en el Anexo #9.

### 3.5.2 Resultado de las pruebas no funcionales

A continuación, se muestran los resultados obtenidos en las pruebas de usabilidad teniendo en cuenta los siguientes indicadores:

1. **Formularios**: deberán reunir los campos adecuados, con la opción de focalizar los campos mediante tabulación y además tenerlos correctamente agrupados, en un orden lógico y alineados, pues la composición visual juega un papel importante. Un formulario con los campos desalineados transmite desorden y genera escepticismo en el usuario que rellena los datos.
2. **Alineamiento**: una composición con pocas líneas de referencia (tanto verticales como horizontales) resulta armónica a la vista del usuario. Por el contrario, una composición con muchas líneas de referencia, escalonada, o con diagonales, tiende a crear desasosiego y sensación de desorden.
3. **Tipografía:** la selección de una u otra tipografía sirve para transmitir valores tan diversos como seriedad, frescura, novedad y antigüedad. El uso aleatorio de tipografías resta seriedad y consistencia a los mensajes.
4. **Colores**: deben aplicarse en las proporciones adecuadas según determinen las guías de estilo. La armonía visual se logra mediante combinaciones de colores que no contrasten excesivamente o que no provoquen fatiga visual.
5. **Retroalimentación al usuario**: durante todo el proceso de instalación se debe indicar la tarea en ejecución y las tareas ya ejecutadas además de notificar al usuario en caso de errores. De esta manera el aprendizaje del instalador es más fácil.




Gráfica 3. Prueba de usabilidad.

Con los datos mostrados en la gráfica tres se puede afirmar que el instalador para el SGM desarrollado es totalmente usable evidenciado mediante los resultados positivos obtenidos.

## 3.6 Conclusiones

- En este capítulo quedaron definidos los estándares de codificación los cuales ayudaron en el entendimiento y posterior uso del código para cualquier programador interesado en este tipo de producto.
- Se elaboró el diagrama de componentes para modelar y documentar la arquitectura del perfil de instalación.
- Se puso en práctica el método de prueba de caja negra para encontrar la mayor cantidad de errores, corregirlos y al mismo tiempo tener un mayor control del software.
- Se utilizó la técnica no probabilística de muestreo intencionado para medir el nivel de esfuerzo empleado en el despliegue del SGM haciendo uso del instalador y sin él.
- El test de usabilidad utilizado permitió comprobar que el instalador desarrollado es usable.

## Conclusiones generales

Después de valorar los resultados obtenidos mediante los métodos y técnicas aplicadas durante la investigación y tras dar cumplimiento al objetivo propuesto con el desarrollo del perfil de instalación para el SGM se establecen las siguientes conclusiones:

- El estudio de las soluciones similares permitió conocer las características comunes entre los instaladores analizados, además se demostró que ninguno de los instaladores analizados tiene como objetivo principal la instalación de un sistema de gestión de eventos científicos.
- El desarrollo del flujo de análisis y diseño correspondiente a la metodología seleccionada, permitió generar los diagramas de clases, diagramas de interacción y diagramas de clases del diseño correspondientes para la posterior implementación.
- La validación de los requisitos funcionales, así como la rectificación de las no conformidades encontradas responden a una elevada satisfacción de los clientes y verifican que la solución es adecuada para el proceso de gestión de eventos independientemente de las reglas de negocio.
- Con el método de caja negra se logró tener un software completamente validado y listo para usar.
- La técnica de muestreo intencionado utilizada demostró que con el uso del instalador desarrollado se minimizó el esfuerzo en cuanto al despliegue del SGM.
- Se demostró mediante un test de usabilidad que el instalador para el SGM desarrollado es usable.

## Recomendaciones

- Continuar con el desarrollo del perfil de instalación para incorporarle más funcionalidades como, por ejemplo: crear perfiles de usuarios.

## Referencias bibliográficas

ADRIAN RAMOS, S.L., 2014. Patrones de Diseño (XV): Patrones de Comportamiento - Command. *Programación en Castellano.* [en línea]. [Consulta: 1 marzo 2016]. Disponible en: http://programación.net/articulo/patrones_de_diseno_xv_patrones_de_comportamiento_command_1018.

ALARCÓN, J.M., 2016. Los 10 mejores editores gratuitos (o casi) de HTML, CSS y JavaScript. *campusMVP.es* [en línea]. [Consulta: 26 enero 2016]. Disponible en: http://www.campusmvp.es/recursos/post/los-10-mejores-editores-gratuitos-de-html-css-y-javascript.aspx.

ALDIBIER MORALES MORALES, 2009. Ejemplo de aplicación sobre Zend Framework - Parte 2: La arquitectura MVC de la aplicación | aldibier.com. [en línea]. [Consulta: 10 marzo 2016]. Disponible en: http://www.aldibier.com/blog/articulo/ejemplo-de-aplicaci%C3%B3n-para-un-blog-sobre-zend-framework-parte-2-la-arquitectura-mvc-de.

ALDIBIER MORALES MORALES, 2010. POO y patrones de diseño en DRUPAL | aldibier.com. [en línea]. [Consulta: 1 marzo 2016]. Disponible en: http://www.aldibier.com/blog/articulo/poo-y-patrones-de-dise%C3%B1o-en-drupal.

Arquitectura | Tutorial Drupal. [en línea], [Consulta: 20 abril 2016]. Disponible en: http://www.cursosdrupal.com/content/arquitectura.

BEASTIEUX ZEROO, 2008. Metodologías para la gestión y desarrollo de Software. [en línea]. [Consulta: 25 febrero 2016]. Disponible en: http://es.scribd.com/doc/8255409/Metodologias-para-la-geston-y-desarrollo-de-Software.

BERNALDO FLORES, 2011. Qué es un entorno de desarrollo integrado, IDE. *Programación Desarrollo* [en línea]. [Consulta: 26 enero 2016]. Disponible en: http://programaciondesarrollo.es/que-es-un-entorno-de-desarrollo-integrado-ide/.

DRIES BUYTAERT, 2013. Drupal 7 - Más fácil y potente que nunca | Drupal.org. [en línea]. [Consulta: 17 febrero 2016]. Disponible en: https://www.drupal.org/drupal-7.0/es.

DRIES BUYTAERT, 2015. Sobre Drupal | Drupal Hispano. [en línea]. [Consulta: 13 diciembre 2015]. Disponible en: http://drupal.org.es/drupal.

DRIES BUYTAERT, M.G., 2009. Requerimientos del sistema | Drupal Hispano. [en línea]. [Consulta: 26 enero 2016]. Disponible en: http://drupal.org.es/node/9.

EVELYN MENDEZ ALONSO, E.M.A., 2014. Herramientas CASE para el proceso de desarrollo de Software - Monografias.com. [en línea]. [Consulta: 26 enero 2016]. Disponible en: http://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software.shtml.

FERNÁNDEZ LADERA, 2012. *CMMI*. 2012. S.l.: s.n. [Consulta: 30 enero 2016].

FRAN GIL RODRÍGUEZ, 2011a. *Experto en Drupal7 nivel avanzado* [en línea]. [Consulta: 30 enero 2016]. 2012 2011. S.l.: s.n. Disponible en: web www.forcontu.com.

FRAN GIL RODRÍGUEZ, 2011b. *Experto en Drupal7 nivel inicial* [en línea]. 2012 2011. S.l.: s.n. [Consulta: 30 enero 2016]. Disponible en: web www.forcontu.com.

GARCÍA, D., 2014. Patrones Estructurales (IV): Patrón Bridge. *Let's code something up!* [en línea]. [Consulta: 1 marzo 2016]. Disponible en: https://danielggarcia.wordpress.com/2014/03/17/patrones-estructurales-iv-patron-bridge/.

GRUPO DE DESARROLLO, 2015. Definición de lenguaje de programación — Definición.de. *Definición.de* [en línea]. [Consulta: 13 diciembre 2015]. Disponible en: http://definicion.de/lenguaje-de-programacion/.

HEREDIA RUIZ, J., RUIZ, J.H., ALMANZA, L.Á. y PONS, N.L., 2011. Comparación y tendencias entre metodologías ágiles y formales. Metodología utilizada en el Centro de Informatización para la Gestión de Entidades. *Serie Científica-Universidad de las Ciencias Informáticas* [en línea], vol. 4, no. 10. [Consulta: 31 enero 2016]. Disponible en: http://publicaciones.uci.cu/index.php/SC/article/view/484.

HONG PONG JAPERRY, 2015. *Desarrollo de perfiles de instalación*. febrero 2015. S.l.: s.n. [Consulta: 31 enero 2016].

IVETTE CAROLINA MARTÍNEZ, 2016. *Pruebas de software*. 2016. S.l.: s.n. [Consulta: 10 febrero 2016].

JACOBSON BOOCH, 2014. *El proceso unificado de software* [en línea]. S.l.: s.n. [Consulta: 10 febrero 2016]. Disponible en: https://es.scribd.com/doc/50327385/El-Proceso-Unificado-de-Desarrollo-de-Software-Jacobson-Booch-Rumbaugh.

JAVASCRIPT COMMUNITY, 2015. JavaScript.com. [en línea]. [Consulta: 14 mayo 2016]. Disponible en: https://www.javascript.com/.

JOSE ANTONI DORADO CERÓN, 2015. Arquitectura de Drupal 8 tras su integración con Symfony2. [en línea]. [Consulta: 1 marzo 2016]. Disponible en: http://www.ladrupalera.com/drupal/desarrollo/drupal8/arquitectura-de-drupal-8-tras-su-integracion-con-symfony2.

LARMAN CRAIG, 1999. *UML Y PATRONES*. S.l.: s.n. [Consulta: 3 marzo 2016].

MANUEL SIERRA, 2015. Qué es y para qué sirve el lenguaje CSS (Cascading Style Sheets - Hojas de Estilo). [en línea]. [Consulta: 26 enero 2016]. Disponible en: http://www.aprenderaprogramar.com/index.php?option=com_content&id=546:que-es-y-para-que-sirve-el-lenguaje-css-cascading-style-sheets-hojas-de-estilo&Itemid=163.

MARTIN FOWLER, 2015. Modelo de dominio. En: Page Versión ID: 84345382, *Wikipedia, la enciclopedia libre* [en línea]. [Consulta: 1 marzo 2016]. [Consulta: 18 marzo 2016]. Disponible en: https://es.wikipedia.org/w/index.php?title=Modelo_de_dominio&oldid=84345382.

MIGUEL MARTÍN ROMO, 2004. *IAN SOMMERVILLE .Séptima edición. Ingeniería de software.* [en línea]. S.l.: s.n. [Consulta 22 marzo 2016]. Disponible en: http://zeus.inf.ucv.cl/~bcrawford/Modelado%20UML/Ingenieria%20del%20Software%207ma.%20Ed.%20-%20Ian%20Sommerville.pdf.

ORACLE CORPORATION, 2016. Bienvenido a NetBeans y www.netbeans.org, Portal del IDE Java de Código Abierto. [en línea]. [Consulta: 14 mayo 2016]. Disponible en: https://netbeans.org/index_es.html.

PHP GROUP, 2001. PHP: Hypertext Preprocessor. [en línea]. [Consulta: 14 mayo 2016]. Disponible en: http://php.net/.

RAUL, 2013. Distribuciones en drupal | drupalia.cat. [en línea]. [Consulta: 13 diciembre 2015]. Disponible en: http://drupalia.cat/tutorial/distribuciones-en-drupal.

ROCHEN, 2015. Joomla! The CMS Trusted By Millions for their Websites. *Joomla!* [en línea]. [Consulta: 16 marzo 2016]. Disponible en: https://www.joomla.org/.

SIRILO JUNO, 2013. Definición de CMS. [en línea]. [Consulta: 13 diciembre 2015]. Disponible en: http://10conceptos.com/definicion-de-cms/.

STEVIA RODRÍGUEZ, 2015. AUP Ingeniería de Software. [en línea]. [Consulta: 13 diciembre 2015]. Disponible en: http://ingenieriadesoftware.mex.tl/63758_AUP.html.

TAMARA RODRÍGUEZ SÁNCHEZ, 2014. *Metodología Aup Uci*. 12 noviembre 2014. S.l.: s.n. [Consulta 30 enero 2016].

VISUAL PARADIG COMMUNITY, 2014. What is Visual Paradigm? [en línea]. [Consulta: 14 mayo 2016]. Disponible en: https://www.visual-paradigm.com/features/.

W3C WORKING GROUP NOTE, 2015. CSS Snapshot 2015. [en línea]. [Consulta: 14 mayo 2016]. Disponible en: https://drafts.csswg.org/css-2015/.

W3C WORKING GROUP NOTE, 2016. W3C HTML. [en línea]. [Consulta: 14 mayo 2016]. Disponible en: https://www.w3.org/html/.